ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Grado en Ingeniería de Computadores

Sistema de monitorización de redes informáticas. Caso real 'Turismo Andaluz'

Computer network monitoring system. The 'Turismo Andaluz' case

Realizado por

Laura González García

Tutorizado por

Julián Ramos Cózar

Departamento

Arquitectura de Computadores

UNIVERSIDAD DE MÁLAGA

MALAGA, Octubre 2.014

Resumen

Este Trabajo Fin de Grado, describe la implantación de un sistema de monitorización de redes informáticas. Se definirán los principales conceptos de monitorización, y se argumentará la elección de la herramienta finalmente seleccionada para llevarlo a cabo. Detallaremos el proceso de instalación, configuración y puesta en producción. Por último, se mostrará cómo funciona el sistema ya instalado sobre la red informática de la Empresa Pública de Turismo Andaluz, S.A., cubriendo las necesidades de control de sistemas desde su sede principal, sita en Málaga, al resto de provincias andaluzas, donde posee diversas sedes secundarias.

Palabras claves:
sistemas, monitorización, red, servidores, servicios, nagios, pandora, cacti, gammu, mrtg, nrpe, nsclient+, notificación, alerta

Abstract

The present work describes the deployment of an ICT network monitoring system. We will define monitoring main concepts and will show the reasoning behind the tool selection process. We will give details about installation, configuration and final deployment of the tool. Finally, the paper will present a real installation on the Empresa Pública de Turismo Andaluz, S.A. network, that supports the users monitoring needs from its main site in Malaga to the satellite sites locate on the other seven Andalusian provinces.

Keywords:
smtp, monitoring, systems, protocols, notification, alert, check_nrpe, check_nt, check_ping

ÍNDICE

1. INTRODUCCIÓN

El desarrollo de este proyecto, surge como una necesidad tras años de experiencia en el Área de Sistemas del Departamento Informático. Día tras día, el número de instalación de servidores, servicios, aplicaciones, dispositivos electrónicos,... va aumentando para que la infraestructura tecnológica de la empresa pueda funcionar correctamente. El tiempo que debe destinarse a confirmar que no existen errores o fallos en cualquier sistema, también crece proporcionalmente.

Por ello, con más o menos criticidad, es necesario habilitar un sistema que se encargue de la monitorización y un sistema de alertas, para que en caso de que se produzca algún problema, se le notifique al personal responsable. De esta forma, los recursos humanos del departamento pueden dedicarse a otras funciones, al ser descargado de esta continua “vigilancia”. Este mecanismo deberá funcionar de forma autónoma en períodos de 24x7 todos los días del año.

1.1. Objetivo

El objetivo principal de este Trabajo Fin de Grado (TFG, a partir de ahora), será la implantación de un robusto sistema de monitorización para la Empresa Pública de Turismo Andaluz, S.A. (TASA, a partir de ahora) que cubra todas las necesidades de control de sus principales sistemas informáticos, electrónica de red y líneas de datos, en las diferentes sedes que tiene actualmente en Andalucía.

Para llevar a cabo este proyecto, se realizará un análisis de las herramientas que existen actualmente, para decidir cuál es la más indicada para cubrir sus necesidades y entonces, realizar una exhaustiva descripción del proceso de instalación y configuración.

A lo largo del desarrollo, se realizará un proceso, de decisión sobre qué dispositivo, servicio o componente es el que realmente nos interesa tener vigilado dependiendo de su naturaleza. Es decir, de un sistema de almacenamiento, deberíamos comprobar el estado de su capacidad en disco, sin embargo, de un servidor web, quizás deberíamos monitorizar que la página que esté mostrando no produzca un "error 500".

Toma de decisiones que se deberán llevar a cabo:

- los dispositivos que deben ser controlados: estudio de los elementos que deben ser vigilados y qué servicios de cada uno de ellos debemos tener en cuenta para ser controlado

- cuáles deben ser los niveles de criticidad para notificar los problemas: se decidirá qué importancia tiene la pérdida de servicio de cada uno de ellos, para crear un patrón de actuación según sea su relevancia.

- a quién se debe enviar el aviso de cada error. Según lo decidido anteriormente, se establecerá a quién le deben de llegar las alertas establecidas, en caso de caída del dispositivo o del servicio

- durante qué períodos de tiempo: decidir en qué horario se deben hacer las notificaciones sobre el cambio de estado de un elemento, una vez más, dependiendo de la importancia de su caída

- de qué forma se debe hacer la notificación: de los diferentes medios proporcionados: email, SMS

- en casos concretos, qué proceso debe ejecutarse para resolver el problema: en aquellos casos en los que el sistema pueda reparar el problema generado,

tras la ejecución de un proceso adicional, también llamado evento, el cual devolverá el servidor/servicio a un estado correcto tras su ejecución

Además, se añadirán aplicaciones complementarias a la herramienta principal de monitorización, que al interactuar entre sí, proporcionarán un total conocimiento de los problemas que puedan producirse y, lo que no es menos importante, qué pasaba antes de ocurrir el problema y facilitar datos adicionales que nos ayuden a comprender mejor el error y la forma de solucionarlo.

Finalmente, se mostrará cómo se ha aplicado todo ello a las instalaciones de TASA y el funcionamiento de la misma sobre su completa red informática.

1.2. Contenidos de la memoria

Esta memoria se encuentra divida en varios capítulos. Se comenzará, por el capítulo denominado “MONITORIZACIÓN”, dedicado a definir el concepto en sí, y conocer qué es lo que se puede y debemos monitorizar en una red informática. Para ello, realizaremos una comparativa entre las principales herramientas existentes, decidiendo cuál es la que mejor se adapta a nuestras necesidades y llevaremos a cabo un estudio más detallado de su funcionamiento.

A continuación, en el el capítulo denominado, “INSTALACIÓN Y CONFIGURACIÓN DE NAGIOS”, se mostrará cómo debe realizarse el proceso de instalación y configuración, de la aplicación en sí. Indicaremos los principales pasos a seguir, desde la descarga de la aplicación para clientes y servidores, hasta la enumeración de los ficheros y directorios implicados en su configuración. Podremos comprobar cómo accedemos a la información de nuestra red desde distintos tipos de dispositivos. Y finalizaremos este capítulo, detallando cómo utilizar aplicaciones adicionales, a modo de complemento para obtener mayor información del estado de nuestra red.

Por último, en el capítulo denominado “CASO REAL EN TURISMO ANDALUZ “ se aplicará todo lo descrito a lo largo de esta memoria en el caso real de la red informática de Turismo Andaluz, S.A. Tras describir las características que definen en concreto esta red, se detallará cómo se realiza el proceso monitorización de los servidores / servicios, y los tipos de chequeos que se realizan dependiendo del dispositivo del que se trate. Describiremos con más detalle, los informes que Nagios puede generar sobre el estado de nuestro sistema. Indicar que las figuras añadidas para ilustrar la información son reales, sin embargo, los datos críticos de las mismas aparecerán difuminados por medidas de seguridad.

En el apartado de “Conclusiones” se indicarán las principales deducciones a las que

nos ha llevado el desarrollo de este proyecto y la redacción de esta memoria, aportando las mejoras deseables e interpretar cuál debería ser la evolución natural de la herramienta en sí.

Al final de este documento, se incluyen las “Referencias bibliográficas”, donde se reúnen los enlaces a las principales fuentes de referencia empleadas y los “Anexos” con información útil y ejemplos para una mejor comprensión de lo descrito a lo largo de la memoria.

Para realizarlo, las fases de trabajo han sido las siguientes:

- Estudio de las principales herramientas de monitorización de software libre disponibles
- Selección y justificación del aplicativo elegido
- Creación de un servidor virtual e instalación los componentes y programas necesarios para el correcto funcionamiento de la herramienta
- Instalación de Nagios en el servidor y en los clientes con sistemas operativos: Linux y Windows
- Configuración de la misma, sus principales ficheros y directorios
- Estudio de las principales opciones y contenidos del interfaz web
- Instalación y configuración de complementos de navegadores web y de aplicaciones móviles para el acceso a las alertas y el estado de la red
- Implementación de todo ello sobre la red de TASA

2. MONITORIZACIÓN

A lo largo de este capítulo, intentaremos definir de forma sencilla y clara, el concepto en sí de monitorización. Justificando porqué se debe contar con este tipo de herramientas en los departamentos de sistemas informáticos que gestionen su propia red. Veremos cuáles son actualmente las algunas herramientas disponibles para ello y tras seleccionar la más apropiada para nuestro caso, entraremos en más detalle de cómo realiza esta función de control de los dispositivos y servicios que se le indique.

2.1. Concepto de Monitorización

La monitorización de sistemas se realiza mediante una aplicación, que bien configurada, analiza de forma constante el estado de los servidores / servicios, durante el tiempo que le indiquemos y que en caso de fallo, podrá avisar a la persona de contacto que deseemos, del tipo de problema surgido.

Al ser avisados a tiempo, los problemas surgidos podrán ser corregidos, evitando incidencias mayores, o incluso, conseguir que cuando se produzca un fallo conocido, se ejecute algún evento que lo corrija sobre la marcha, sin necesidad de alguna acción por parte del personal responsable.

Como se puede deducir, además de la descarga de trabajo que supone no tener que estar comprobando siempre el estado de todos los sistemas, este aplicativo proporciona seguridad y tranquilidad a las personas implicadas, ya que en este caso se cumplirá la premisa: “Si no hay noticias, son buenas noticias”, es decir, mientras los responsables no reciban notificaciones de la herramienta de control, pueden dedicarse a otros menesteres, ya que lo monitorizado está funcionando correctamente.

Este control del estado de los objetos monitorizados se llevará a cabo mediante el intercambio de información entre un servidor central y el resto de sistemas a vigilar. El diálogo se realizará a través de protocolos de comunicación entre los sistemas implicados. A partir de esta información y según se haya indicado en la configuración, qué es lo que debe interpretarse como un error, el servidor principal, deducirá si existe algún problema y qué acción debe llevar a cabo: enviar un email, un SMS, ejecutar alguna acción que corrija el problema.

Para llevar a cabo esto, algunos sistemas necesitarán la instalación de algún tipo de software adicional (cliente) para que pueda “hablar” correctamente con el servidor principal y facilitarle la información necesaria, para evaluar si todo está funcionando correctamente. Esto dependerá de lo que deseemos monitorizar en el sistema remoto. No es lo mismo comprobar si un sistema está encendido, que saber la cantidad de memoria RAM que tiene disponible un sistema para trabajar de forma óptima.

2.2. ¿Qué debemos monitorizar?

La premisa de la que debemos partir para tomar esta decisión sería: “De qué componentes de mi sistema debería recibir un aviso en caso de que exista algún problema, de tal forma que no deba estar comprobando yo mismo su estado continuamente”. Es decir, aquello que si falla, quiero saberlo cuanto antes para que el problema no derive en otro mayor.

Como la naturaleza de todos los componentes de un sistema informático / electrónico es muy distinta, también deberemos tener en cuenta qué aspectos son más críticos que otros. Por ejemplo, desearía recibir un SMS cuando haya una caída en la línea principal de la sede central de la empresa, para avisar a nuestro proveedor de servicio lo antes posible, sin embargo, para el caso en que el espacio libre que tiene un sistema de archivos, dedicado a las copias de seguridad, ha bajado del 20%, con recibir un email de notificación, tendría suficiente y corregirlo una vez esté físicamente en la oficina.
Podemos enumerar algunos de los principales objetos que deberían ser monitorizados:

- Todos los servidores principales: Firewall, Proxy, Correo Electrónico, DNS

- Todos los servidores de web

- Todos los switches pertenecientes a la electrónica de red. En concreto los puertos por
- donde debamos asegurarnos que existe tráfico de información

- Todos los routers de líneas de datos (tanto principales como de backups)

Y de ellos, deberíamos asegurarnos de recibir información en caso de problemas con:

- Espacio en disco / sistemas de archivos
- Gestión de memoria
- Número de procesos activos
- Número de procesos “zombies”
- Existencia de ficheros: Tanto de copias de seguridad, como de logs
- Nivel de carga en CPU
- Estado de puertos en servidores, para confirmar que realmente están “escuchando”
- Inodos de los sistemas de archivos con mayor volumen de creación de ficheros
- Caducidad de certificados web de dominios publicados bajo https
- Tiempo de actividad del sistema

2.3. Herramientas de Monitorización

Actualmente, existen numerosas herramientas software que se encargan de la monitorización de sistemas.

Para llevar a cabo un comparativo entre ellas, nos basaremos en la documentación que nos ofrece la Wikipedia. Si accedemos a ella, podremos encontrar un gráfico, que por sus dimensiones no es posible mostrar aquí, en la que se enumeran todas las herramientas de monitorización que existen, con todas las características que poseen. El enlace a esta imagen es:

Referencia a cuadro resumen de comparativo de herramientas:
http://es.wikipedia.org/wiki/Anexo:Comparaci%C3%B3n_de_sistemas_de_monitorizaci%C3%B3n_de_redes

Por simplificar este comparativo, en nuestro caso, sólo buscaremos entre aquellas herramientas, que siendo OpenSource (Código Abierto, por tanto, sin licencia ni coste económico) y totalmente configurable, nos proporcione los medios necesarios para poder controlar la mayoría de los sistemas. Por no ser motivo de este proyecto, el estudio detallado de dichas herramientas, se realizará una enumeración de las más importantes, indicando sus ventajas e inconvenientes de forma breve.

Como se mencionó anteriormente, el desarrollo de este TFG se realizará con una sola de ellas, elección que justificaremos llegado el momento.

2.3.1. Comparativa de herramientas

Las herramientas de control de redes se diferencian unas de otras, de cómo realizan el intercambio de información con los dispositivos, la forma en la que la gestionan, la representan y, en caso de que lo hagan, cómo la transforman en alertas para los

usuarios. Por ello, destacaremos de las principales aplicaciones de monitorización, los aspectos que la hacen destacar frente a otras y las carencias que presenta.

Principalmente, las aplicaciones a evaluar deberán de poder facilitarnos información acerca de cualquier tipo de dispositivo que podamos tener dentro de nuestra red informática y de la mayoría de sus componentes en tiempo real: servidores físicos, virtuales, servicios de correo, antivirus, certificados web que caduquen, tráfico en bocas de routers o switches, sensores de funcionamiento, dispositivos wifi, espacio en dispositivos de almacenamiento (NAS), etc. Teniendo en cuenta la continua evolución a la que están sometidas todas las redes, se tendrá también en cuenta, que las herramientas deben tener la capacidad de adaptarse a este proceso evolutivo. Además, de mostrarlo gráficamente en un intuitivo interfaz y poseer mecanismos para avisarnos en caso de problemas.

Veamos un resumen de las principales herramientas que existen en la actualidad:

Pandora FMS

Es de los sistemás más jóvenes y se ha convertido en una de las aplicaciones más populares y completa. Desarrollada por una empresa española: Ártica. Es una herramienta que puede estar indicada para grandes instalaciones. A su favor, podemos indicar que puede monitorizar, básicamente, cualquier tipo de sistema operativo y casi cualquier tipo de dispositivo o servicio. Esto puede realizarlo teniendo el sistema remoto instalado un cliente o a través de protocolos, sin tener que instalar nada en el destino. Puede enviar notificaciones vía email o SMS en caso de detección de problema. Necesita un gestor de base de datos, por defecto MySQL, donde almacena toda la información para ser procesada.
Posee un interfaz web desarrollado en PHP5 muy ligero que, sin embargo, genera las gráficas en Flash, por lo que los navegadores deberán tener implementado este complemento para visualizarlas. Su configuración puede gestionarse desde su interfaz web, por lo que no necesitará alguien demasiado experto en sistemas

informáticos para su mantenimiento, una vez haya sido correctamente instalada y puesta en funcionamiento.

En su forma de trabajo, consigue dividir en varios procesos la ejecución de sus tareas, haciendo un mejor uso de los recursos. Los cambios en su configuración, no implican un reinicio del global del sistema.

Pandora FMS ha desarrollado su propio protocolo de transferencia de información, Tentacle, para intercambiar información con los clientes en los sistemas remotos, que quizás consuma más de lo deseado de recursos de tipo entrada/salida.

Al ser relativamente nueva, la comunidad que la soporta no es aún muy numerosa, por lo que el mantenimiento de plugins y creación de nuevos no es demasiado dinámica.

Aunque con la versión OpenSource quizás se pueda cubrir la mayoría de las necesidades de cualquier red informática, además, posee una versión bajo licencia. En la **Figura 2.1** (fuente http://www.pandorafms.org), vemos su imagen comercial.

Figura 2.1. Logo de Pandora FMS

Nagios

También muy popular y consolidada, durante años viene siendo la herramienta de

monitorización por excelencia. Reúne las principales ventajas nombradas en el caso anterior, si bien, para Nagios, se necesitará que sea el personal del departamento de sistemas el que gestione la instalación y configuración en el servidor, así como, los clientes necesarios en los dispositivos remotos.

Desarrollada en Perl, integra un intérprete para los plugins desarrollados en este lenguaje, consiguiendo aumentar su rendimiento.

Uno de sus puntos fuertes de es la gran flexibilidad en su configuración, así como, la creación de nuevos plugins que se encarguen de monitorizar “algo” para lo que todavía no existía ninguna opción o realizar alguna modificación a algún plugin existente para adaptarse a nuestras necesidades. Éstos pueden ser programados en diversos lenguajes como Perl, C, PHP, Python, etc. Existe una gran comunidad detrás de esta herramienta que la hace estar en continúa evolución y actualizada.

Se pueden establecer jerarquías entre los dispositivos, establecer períodos diferentes de monitorización según las necesidades existentes, diferentes plantillas para controlar un mismo recurso, según sea la naturaleza del dispositivo o servicio a controlar.

También permite las notificaciones por correo electrónico o SMS, diferenciando los niveles de criticidad.

Como desventaja, habría que destacar su bajo nivel de gráficas de estado, probablemente debido a no soportarse bajo un motor de base de datos que almacene toda la información. Además, los cambios que realicemos en cualquier fichero de configuración, necesitarán un reinicio del servicio para que pueda ser aplicado. Su arquitectura se basa en un único proceso, que ejecuta todas las tareas programadas.

Hay una variante comercial de este producto, Nagios XI, basándose en el volumen

de nodos a monitorizar. En la **Figura 2.2** (fuente http://www.nagios.org), se muestra su imagen comercial.

A partir de ella se han generado otras herramientas, manteniendo el núcleo de Nagios como son: Opsview o Shinkem

Figura 2.2. Logo de Nagios

OpenNMS

Es una de las herramientas más antiguas, pero que ha seguido evolucionando hasta hoy día. Creada también para monitorizar casi todos los tipos de dispositivos de una red, al igual que en los otros casos nombrados, posee una versión totalmente gratuita alimentada por la comunidad que la mantiene y otra versión comercial.

Puede lanzar aplicaciones o scripts, cuando detecta un cambio de estado en los dispositivos / servicios, incluso las notificaciones pueden ser enviadas a otras herramientas para centralizar toda la información de estado.

De las funciones más importantes que añade esta aplicación, es la de ser capaz de detectar nuevos servicios y añadirlos a su configuración, aspecto muy útil para el personal responsable, ya que también le descarga de tener que añadirlo a la configuración una vez instalado. Aunque también dispone de la posibilidad de hacerlo manualmente.
Genera sus propios informes y gráficas, de los datos almacenados en su base de datos en PostgreSql (único motor que soporta). Se encuentra desarrollada en Java y soporta plugins diseñados para Nagios, que actualmente es el que más plugins genera.

La instalación y, sobre todo, posterior configuración para obtener un nivel de rendimiento óptimo necesita de altos niveles de conocimiento, además, su interfaz web no es demasiado intuitivo y puede no ser compatibles con todos los navegadores actuales. En la **Figura 2.3** (fuente http://www.opennms.org) se muestra su imagen comercial.

Figura 2.3. Logo de OpenNMS

Zabbix

Otra importante opción a tener en cuenta es Zabbix. Monitoriza las redes utilizando clientes o mediante protocolos, para poder controlar aspectos como carga del dispositivo, actividad en la red, parámetros del sistema operativo, etc. Generando informes y gráficas en su interfaz web desarrollada en PHP y JavaScript. Su servidor y agentes están desarrollados en lenguaje C. Soporta casi todos los motores de bases de datos: MySQL, PostgreSQL, SQLite, Oracle o IBM DB2.

Posee un sistema de alertas para envío de notificaciones a través de email o SMS, pero además añade el uso de XMPP (antes llamado Jabber).

Posee la ventaja de detectar nuevos elementos añadidos a la red y puede establecer una jerarquía entre ellos.

Sin embargo, la capacidad de esta herramienta viene limitada a un número total de 1.000 nodos y al no poseer una versión Enterprise hace que no esté a la misma altura que otras herramientas de características similares.

En la **Figura 2.4** (fuente http://www.sudoers.com) se muestra su imagen comercial.

Figura 2.4. Logo de Zabbix

Cacti

Al igual que el resto de aplicaciones anteriormente referidas, recopila información de los sistemas remotos para controlarlos, en este caso, sólo a través SNMP. Además, presenta una importante gestión de las gráficas con los datos almacenados desde el momento en que empezó a monitorizar el sistema o servicio. A través de su interfaz web se realiza su configuración y permite, por ejemplo, realizar un comparativo entre todas las cargas de CPU de los servidores, o bien, comprobar el estado de espacio en disco de los servidores pertenecientes a una sede. También necesita un gestor de base de datos, MySql, para almacenar la información. Gestiona un importante sistema para definir plantillas y adecuar el interfaz gráfico a nuestras necesidades de información.

No posee clientes para instalar en los servidores remotos y así, poder obtener información de otros parámetros adicionales que pudieran ser necesarios monitorizar. Tampoco posee un sistema de alertas de envío en caso de errores. En la **Figura 2.5** (fuente http://datalibre.blogspot.com.es/) vemos su logotipo.

Figura 2.5. Logo de Cacti

2.3.2 Herramienta seleccionada

¿Cuál de las opciones es la mejor? La mayoría de las herramientas que hemos visto, cumple ampliamente las necesidades que puedan surgir en cualquier red informática a monitorizar. Así que para seleccionar una de ellas, se decidió tomar la decisión desde otro punto de vista, como valorar aquella que se adaptase de la forma más óptima posible, a la red en concreto de TASA, en la que finalmente será instalada y objeto de este TFG. Teniendo muy en cuenta, los recursos materiales y humanos disponibles para ello y la futura explotación y evolución de la misma, según el crecimiento de esta red.

Una vez hemos realizado un comparativo con las principales aplicaciones y tenido en cuenta lo anteriormente mencionado, debemos decidirnos por una de ellas. La elección en nuestro caso ha sido **Nagios**.

El personal del departamento de sistemas de TASA, está ampliamente cualificado para realizar el tipo de instalación, configuración, puesta en marcha y mantenimiento de este tipo de herramientas, que ellos mismos explotarán. Es decir, los usuarios finales también serán casi siempre ellos.

De este sistema aprovecharemos todas sus ventajas y en la medida de lo posible, supliremos sus desventajas de la mejor manera posible:

- Flexibilidad: de cada servidor monitorizaremos sólo lo que nos interesa, podemos agrupar muchos servidores o servicios para una sola entrada de chequeo.

- Diferenciar para un mismo plugin de control, distintos tipos de alertas basándonos en su criticidad: Por ejemplo, necesitamos que el chequeo del ping del cortafuegos nos avise en cuanto aumente la latencia, sin embargo, el

mismo control para una impresora o un dispositivo wifi, podremos asignarles umbrales mayores usando el mismo plugin.

- A través de la configuración, haremos que el interfaz gráfico nos represente la información de forma muy intuitiva, con agrupaciones de hosts o servicios por ubicación o naturaleza, o bien, generando distintos tipos de representación jerárquica de todos los elementos que la forman.

- Aprovecharemos la integración con otras aplicaciones para aumentar la información y mejorar las gráficas que presenta Nagios por defecto.

- Existen muy diversas formas de configurar, agrupar, invocar a los plugins. Ya sea a través de protocolos o mediante instalación de clientes en el servidor remoto.

- A través del servidor web y mediante validación con nuestro LDAP, mejoraremos las medidas de seguridad para poder acceder a la herramienta, que el que trae por defecto.

- Los aportes que realiza la comunidad a esta herramienta, hace que podamos monitorizar cualquier dispositivo o servicio, ya que los plugins se van creando a medida que van apareciendo nuevas demandas. Además, podremos adecuarlos y personalizarlos a nuestra red de forma muy sencilla modificando el código de los propios plugins.

- Para el personal de sistemas, especializado en entornos Linux, se prefiere realizar el mantenimiento sobre los ficheros configuración y tener bajo control todo lo que se está modificando, en lugar de realizarlo vía interfaz web.

- Con la amplia variedad que soporta de servicios de red a monitorizar, las

necesidades quedan sobradamente cubiertas: SMTP, POP3, HTTP, ICMP, SNMP...

- Al poder establecer una jerarquía, el sistema de monitorización podrá diferenciar si el sistema/servicio que vigila está realmente caído o que no puede acceder a él, porque hay un objeto superior que se lo impide.

- Las notificaciones en caso de alerta también son de fácil y flexible configuración, pudiendo además, establecer jerarquías, que en caso de no resolverse un error, pueda enviar la alerta a alguien con mayor poder de decisión y gestionar la solución del problema.

- Uno de los aspectos más valorados es la posibilidad de llevar a cabo acciones por sí solo para solucinar algún problema en caso de producirse. Lanzamientos de eventos que veremos más adelante de forma detallada.

2.3.3. Funcionamiento de Nagios

Veamos gráficamente, cómo funciona la aplicación de monitorización seleccionada, Nagios, en la **Figura 2.6** (fuente http://es.wikipedia.org/).

En ella, se muestra cómo sería el flujo de información a través de Nagios. Por una parte, en la columna de la izquierda aparecerían todos los objetos susceptibles de ser monitorizados: Bases de datos, electrónica de red, servidores web, estado de la memoria, de usuarios conectados, etc.... que transmitirían la información a través de protocolos de comunicación o con la ayuda de clientes instalados en los servidores remotos.

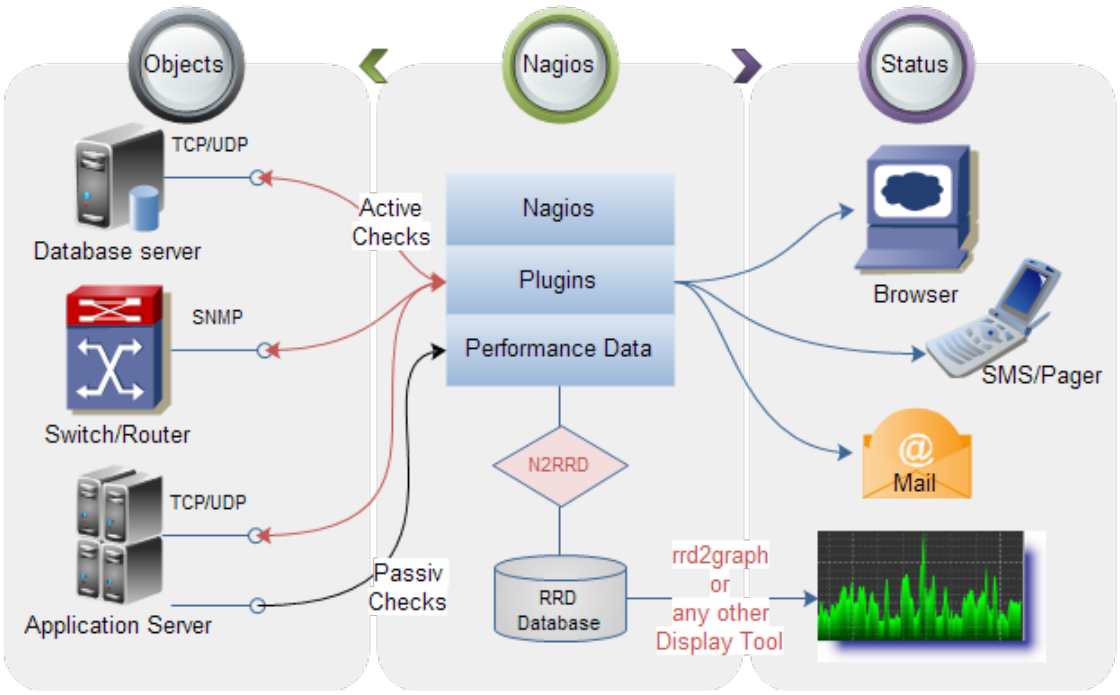


Figura 2.6. Esquema de funcionamiento de Nagios

En la parte central, estaría el servidor de Nagios propiamente dicho, que una vez recibida e interpretada la información, haría accesible la información a través de sus interfaz web o bien, enviaría las alertas por las vías configuradas, (columna de la derecha del gráfico, información del estado).

Adicionalmente, con el complemento N2RRD (Nagios to Round Robin Database), puede almacenar información obtenida y mostrarla en formato de base de datos tipo RRD, que podrá enviar a distintos intérpretes para generación de gráficas (también puede añadirse el complemento rrd2graph).

Para entender la forma de trabajar de Nagios, se debe indicar que una vez recibe la información de los dispositivos y la ha analizado, clasificará el estado de los objetos controlados según los umbrales que le hayamos indicado en:

- **OK**: Todo está funcionando correctamente.

- **WARNING**: Se debería revisar el problema pero hay margen de actuación aún.

- **CRITICAL**: Hay que acudir a solventar el problema lo antes posible, para evitar que pueda derivar en algún problema mayor.

- **UNKOWN**: No recibo información del objeto. Puede que sea porque hay algún dispositivo intermedio que me impida llegar a él o porque no puedo procesar la información que el cliente me está enviando.

En la misma configuración para cada objeto, le indicaremos qué es lo que debe hacer una vez se ha producido un cambio en el estado de los objetos. Así, ante un estado CRITICAL para una servidor PROXY, deberá realizar envíos de email y SMS a los responsables, sin embargo, ante un WARNING de algún parámetro concreto de un servidor, podemos esperar a ver si se corrige por ser algo temporal. Del mismo modo, cuando un objeto vuelva a estar en estado OK, tras haber tenido alguna alerta, también nos lo notificará.

Otro de los aspectos más importantes de Nagios es la posibilidad de lanzar eventos cuando así se lo indiquemos. De forma sencilla, y siempre a través de su configuración para automatizar el mecanismo, consiste en decidir sí al producirse un cambio de estado en un objeto, se debe invocar a un programa, código o ejecución de varios comandos, que realicen una acción determinada.

Quizás con un ejemplo se entienda mejor. Tenemos una aplicación web que por algún motivo deja de funcionar correctamente y el equipo de desarrollo no termina de identificar, porqué llegado un determinado momento, presenta un “Error 500”, es decir, indicará al usuario de la aplicación que esa página no existe, cosa que no es cierta. Es entonces cuando Nagios tras detectar el cambio de estado en el servidor web, enviará las correspondientes notificaciones a los responsables de desarrollo,

pero además, puede lanzar ese evento o código, que haga que el servidor web presente una página de mantenimiento temporal, mientras se corrige el error o consiguen devolver al servidor web a un estado correcto.

Otro caso podría ser un dispositivo con webcam, que tras detectar movimientos realice capturas de imágenes y las almacene en un directorio. Nagios podrá detectar que existe un fichero nuevo y enviar alertas a los responsables, además de lanzar un evento que cifre los ficheros y las envíe a otro servidor como medida de seguridad para su posterior análisis.

Estos son casos muy básicos, pero bastante demostrativos de la importancia de esta característica de Nagios, porque como se puede pensar, se abre un mundo de posibilidades a la hora de intentar corregir o paliar problemas que se produzcan siendo éstos de cualquier naturaleza, y aplicable a la mayoría de dispositivos o servicios.

3. INSTALACIÓN Y CONFIGURACIÓN DE NAGIOS

Éste es sin duda, el capítulo principal de este proyecto. En él se detallará las siguientes etapas: proceso de instalación del servidor, instalación de Nagios, tanto en el cliente como en el servidor, así como, del envío de alertas (por email o por SMS). Se describirán los principales directorios y ficheros para realizar posteriormente su configuración y una vez finalizado, cómo podremos acceder de forma más amigable al estado de nuestra red en diferentes tipos de dispositivos. Se añadirán aplicaciones que nos proporcionen más información a cerca de lo que estaba ocurriendo hasta el momento en el que se hayan generado las alertas. Con todo ello seremos capaces de monitorizar cualquier tipo de red con Nagios de forma sencilla gracias a la flexibilidad para su configuración que proporciona.

3.1. Tipo de servidor y Sistema Operativo seleccionado

Como se refirió al inicio de esta memoria, la finalidad de este TFG es la implantación de un sistema de monitorización para la red informática en el caso de TASA. Por lo que la elección del servidor y su sistema operativo, va a venir condicionado por los recursos existentes y la política del departamento informático de la empresa.

Desde hace años, TASA, siguiendo directrices de la Junta de Andalucía, ha apostado por el software libre y la virtualización de sistemas. Es por ello, que teniendo ya una infraestructura creada y estable, se decida realizar la instalación de Nagios partiendo de instalaciones, sobradamente contrastadas y que nos van a proporcionar un alto rendimiento y fiabilidad.

Por todo esto, se realizará sobre un Contenedor Virtual (también denominado vz) del sistema de virtualización OpenVZ. El principal argumento para justificar su instalación en este tipo de servidor, sería que todos los parámetros iniciales

asignados (espacio en disco, memoria, etc), podrán ser actualizados en “caliente”, cuando esté el servidor ya funcionando, por lo que es preferible comenzar con una configuración ajustada a nuestras necesidades, pero con la tranquilidad de que en el caso de que lo necesitemos, ejecutando un comando en el servidor físico, aumentaremos los recursos del servidor virtual de forma inmediata, sin reinicios ni paradas de servicios alguno.

Además el sistema de copias (creación diaria de dumps), ya establecido sobre el servidor físico, para todos los servidores virtuales que contiene, hace que se realice también sin pérdidas de funcionamiento. Otra de las principales ventajas por la que se ha decidido el empleo de la virtualización es la movilidad. En caso de fallo del servidor físico que la alberga, si debe ser reiniciado o debemos realizar algún tipo de mantenimiento, del mismo modo que antes en “caliente”, podremos mover el contenedor a otro servidor con Openvz, y tampoco habrá parada del servidor virtual.

Finalmente, debemos referir la opción de duplicar un sistema de monitorización partiendo de una copia (o dump) y actualizando su dirección IP, para evitar los conflictos de red. Algo muy útil en caso de querer tener un entorno de preproducción o desarrollo de pruebas.

Del servidor físico en sí, no necesitamos conocer de forma exhausta todos sus recursos, pero sí asegurarnos que el contenedor virtual que vamos a crear, va a poder ejecutarse de forma correcta. Debemos tener en cuenta, que como se ha mencionado anteriormente, el servidor virtual puede ser desplazado de uno a otro físico sin detenerse, por lo que no existe una dependencia del servidor donde se cree. Obviamente, estaremos seguros de que todos los servidores físicos con OpenVZ, que puedan albergar los servidores virtuales, tienen recursos de sobra para ello, pero bueno, de eso ya se encargará Nagios cuando esté funcionando.

Además, se comprobará que posee doble fuente de alimentación, conectadas cada una de ellas a una fase distinta del SAI principal, para evitar caídas por falta de

alimentación eléctrica.

Como tampoco es objeto de este TFG el apasionante mundo de la virtualización de servidores, no quisiera desaprovechar la ocasión para recomendar acceder a http://openvz.org/ y comprobar todo el potencial de la virtualización, con sus ventajas y sus defectos.

Llegado este momento, el sistema operativo seleccionado es un secreto a voces, ya que, la herramienta seleccionada es Nagios, el servidor virtual es de OpenVz y hemos apostado por Opensource. Sí, el sistema operativo será Linux y en concreto, Centos 6.

Sin más preámbulos, vamos a proceder a la creación del servidor virtual para poder instalar Nagios.

3.1.1. Instalación servidor virtual

A continuación, se detalla la creación de un servidor virtual dentro de un servidor físico actualmente en producción.

Para ello, partiremos de una plantilla básica proporcionada por OpenVZ sobre Centos 6 y con los parámetros mínimos que debe poseer el contenedor que albergue a nuestra herramienta, Nagios.

En el servidor físico, vamos a crear el nuevo servidor virtual y los comandos para su creación son relativamente sencillos:

[miservidor]# *vzctl create ID --ostemplate centos-6-x86_64 --config basic*

Con este comando se crea el servidor virtual desde la plantilla indicada (elegimos

Centos 6 de 64 bits), la configuración que va a tener inicialmente (en este caso la básica), así como, el ID que le vamos a asignar, ID (debe ser un valor numérico). Con este paso ya se habría creado el contenedor virtual y ahora deberemos configurarlo y especificar los recursos de los que podrá hacer uso del servidor físico:

> ***[miservidor]#*** *vzctl set ID --hostname vzID.midominio.org --ipadd 192.168.10.3 --nameserver 192.168.10.20 --onboot yes --diskspace 10G:12G --cpuunits 73165 --cpulimit 0 --userpasswd root:xxxx --name NombreServidor --capability sys_time:on –save*

Una vez hemos creado el contenedor virtual, con este comando, le proporcionamos los parámetros de configuración básicos que deseamos, como son: hostname, dirección IP, parámetros de la CPU (con *cpuunits*, le indicamos la garantía de asignación de CPD que le vamos a asignar a nuestro contenedor y con *cpulimit*, el porcentaje máximo de CPU), servidor de nombres, espacio en disco, etc. Especificando, que en esta ocasión, se le han asignado 10GB de espacio físico en disco y que la plantilla básica de la que partimos, le asigna 3GB de memoria RAM. Estos parámetros nos indican que una herramienta tan potente como es Nagios, no necesita grandes recursos para su correcto funcionamiento.

¡Y ya está! Una vez creado nuestro servidor, podremos acceder a él una vez lo arranquemos con el siguiente comando:

> ***[miservidor]#*** *vzctl start ID*

El acceso podrá realizarse a través de un nuevo terminal SSH o desde la misma línea de comando del servidor físico:

> ***[miservidor]#*** *vzctl enter ID*

Este servidor, a partir de ahora, entrará a formar parte del sistema de copias diarias

(backups) para tener la seguridad de no perder información en ningún momento.

3.1.2. Configuración servidor virtual

Una vez accedamos al servidor virtual, trabajaremos igual que si fuera uno físico recién instalado, y al que deberemos hacer algunos cambios, como por ejemplo, establecer una nueva contraseña para el usuario “root”, configurar zona horaria para la correcta sincronización de los sistemas e instalar un repositorio para la descargar e instalación de paquetes y realizar una actualización general del sistema.

```
[miservidor-virtual]# passwd
[miservidor-virtual]# ln -sf /usr/share/zoneinfo/Europe/Madrid  /etc/localtime
[miservidor-virtual]# rpm -Uhv
http://apt.sw.be/redhat/el5/en/x86_64/rpmforge/RPMS/rpmforge-release-0.3.6-
1.el5.rf.x86_64.rpm
[miservidor-virtual]# yum update
```

Además, deberemos eliminar algunos servicios que vienen por defecto como son el servidor de correos “sendmail” y el “iptables”, que no vamos a necesitar en este caso. Siendo también eliminados del proceso de arranque.

```
[miservidor-virtual]# service sendmail stop
[miservidor-virtual]# service iptables stop
[miservidor-virtual]# chkconfig sendmail off
[miservidor-virtual]# chkconfig iptables off
```

Para finalizar, se dejan configurados el servidor web (Apache), que ya viene por defecto, y un servidor de correos (Postfix), necesarios para el funcionamiento de Nagios.

[miservidor-virtual]# *yum install postfix*

Loaded plugins: fastestmirror

Determining fastest mirrors

…. …. ….

Setting up Install Process

Resolving Dependencies

--> Running transaction check

---> Package postfix.x86_64 2:2.6.6-6.el6_5 will be installed

--> Finished Dependency Resolution

Dependencies Resolved

==========

Package Arch Version

Repository Size

==========

Installing:

postfix x86_64 2:2.6.6-6.el6_5 updates

2.0 M

Transaction Summary

==========

Install 1 Package(s)

Total download size: 2.0 M

Installed size: 9.7 M

Is this ok [y/N]: y

Downloading Packages:

postfix-2.6.6-6.el6_5.x86_64.rpm | 2.0 MB 00:04

Running rpm_check_debug

Running Transaction Test

Transaction Test Succeeded

Running Transaction
Installing : 2:postfix-2.6.6-6.el6_5.x86_64
1/1
Verifying : 2:postfix-2.6.6-6.el6_5.x86_64
1/1
Installed:
postfix.x86_64 2:2.6.6-6.el6_5
Complete!

Una vez configurados losparámetros del servicio de correos, sólo quedará iniciarlo y añadir a la lista de procesos que el servidor debe iniciar al arrancar.

[miservidor-virtual]# *service postfix start*
[miservidor-virtual]# *chkconfig postfix on*

3.1.3. Instalación dispositivo envío SMS

Llegado este momento, sólo faltaría una instalación complementaria para realizar los posibles envíos por SMS, en caso de notificaciones por esta vía. Para ello, se utiliza un Modem GSM con una tarjeta SIM y un software que gestione el dispositivo.

Figura 3.1. Kit Modem GSM (fuente http://wiki.ispadmin.eu)

Es un mecanismo muy sencillo, que consta de la instalación de un componente físico formado por los elemnetos mostrados en la **Figura 3.1**:

Su configuración es básica y consta de dos pasos. Por una parte, una vez conectado al servidor físico que contiene a nuestro servidor virtual a través de su puerto serie, se deberá añadir a la configuración del servidor virtual con un simple comando:

[miservidor]# *vzctl set ID --**devnodes ttyS0:rw** --save*

lo cual creará en el fichero de configuración, ID.conf, de nuestro servidor una entrada de este tipo:

….
DEVNODES="ttyS0:rwq "
….

Una vez más, comentar, que este comando se ejecutaría estando el servidor virtual funcionando y sin tener que reiniciar ningún tipo de servicio. De este modo, ya estaría el dispositivo conectado al puerto serie del servidor físico, disponible para ser usado por el virtual.

Ahora sólo debemos instalar el software que lo va a gestionar desde nuestro servidor virtual, en este caso, ya que el tipo de software que gestiona un equipo GSM es muy básico, se ha optado por GAMMU, una vez más, elegimos una aplicación tipo OpenSource y siendo de los más populares y empleado para este fin:

[miservidor]# *yum install gammu.x86_64*
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
** base: sunsite.rediris.es*
** epel: ftp.pbone.net*

** epel-debuginfo: ftp.pbone.net*

** epel-source: ftp.pbone.net*

** extras: sunsite.rediris.es*

** rpmforge: apt.sw.be*

** updates: sunsite.rediris.es*

Setting up Install Process

Resolving Dependencies

--> Running transaction check

---> Package gammu.x86_64 0:1.11.0-1.el6.rf will be installed

--> Processing Dependency: libbluetooth.so.3()(64bit) for package: gammu-1.11.0-1.el6.rf.x86_64

--> Running transaction check

---> Package bluez-libs.x86_64 0:4.66-1.el6 will be installed

--> Finished Dependency Resolution

Dependencies Resolved

==========

Package Arch Version Repository Size

==========

Installing: gammu x86_64 1.11.0-1.el6.rf rpmforge 1.3 M

Installing for dependencies:

bluez-libs x86_64 4.66-1.el6 base 75 k

Transaction Summary

==========

Install 2 Package(s)

Total download size: 1.4 M

Installed size: 4.0 M

Is this ok [y/N]: y

Downloading Packages:

(1/2): bluez-libs-4.66-1.el6.x86_64.rpm

| 75 kB 00:03

(2/2): gammu-1.11.0-1.el6.rf.x86_64.rpm

| 1.3 MB 00:02

==========

Total

217 kB/s | 1.4 MB 00:06

Running rpm_check_debug

Running Transaction Test

Transaction Test Succeeded

Running Transaction

Installing : bluez-libs-4.66-1.el6.x86_64

1/2

Installing : gammu-1.11.0-1.el6.rf.x86_64

2/2

Verifying : bluez-libs-4.66-1.el6.x86_64

1/2

Verifying : gammu-1.11.0-1.el6.rf.x86_64

2/2

Installed:

gammu.x86_64 0:1.11.0-1.el6.rf

Dependency Installed:

bluez-libs.x86_64 0:4.66-1.el6

Complete!

En este momento ya dispondremos de nuestro servidor virtual, arrancado y configurado para instalar el software necesario para monitorizar los sistemas que deseemos.

3.2. Instalación de Nagios

Una vez se ha tomado la decisión de instalar Nagios como herramienta de monitorización, debemos proceder a instalarlo tanto en el servidor principal (servidor virtual creado “ad hoc” para tal fin), como en los clientes remotos de los que necesitemos información de su funcionamiento.

Por tanto, deberemos diferenciar entre el software del servidor y el de cada cliente.

3.2.1. Nagios en el Servidor

La instalación de la aplicación Nagios en sí, sobre el servidor virtual creado, comienza por la descarga de los paquetes y dependencias:

```
[miservidor-virtual]#  yum install nagios nagios-plugins

Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ftp.cica.es
 * epel: ftp.cica.es
 * extras: ftp.cica.es
 * rpmforge: apt.sw.be
 * updates: ftp.cica.es
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package nagios.x86_64 0:3.5.1-1.el6 will be installed
--> Processing Dependency: user(nagios) for package: nagios-3.5.1-
1.el6.x86_64
--> Processing Dependency: user(nagios) for package: nagios-3.5.1-
```

1.el6.x86_64

--> Processing Dependency: nagios-common for package: nagios-3.5.1-1.el6.x86_64

--> Processing Dependency: group(nagios) for package: nagios-3.5.1-1.el6.x86_64

--> Processing Dependency: group(nagios) for package: nagios-3.5.1-1.el6.x86_64

--> Processing Dependency: libgd.so.2()(64bit) for package: nagios-3.5.1-1.el6.x86_64

---> Package nagios-plugins.x86_64 0:1.4.16-10.el6 will be installed

--> Running transaction check

---> Package gd.x86_64 0:2.0.35-11.el6 will be installed

---> Package nagios-common.x86_64 0:3.5.1-1.el6 will be installed

--> Finished Dependency Resolution

Dependencies Resolved

==

Package Arch Version Repository Size

==

Installing:

nagios x86_64 3.5.1-1.el6 epel 1.2 M

nagios-plugins x86_64 1.4.16-10.el6 epel 200 k

Installing for dependencies:

gd x86_64 2.0.35-11.el6 base 142 k

nagios-common x86_64 3.5.1-1.el6 epel 17 k

Transaction Summary

==========

Install 4 Package(s)

Total download size: 1.5 M

Installed size: 6.8 M

Is this ok [y/N]: y

Downloading Packages:

(1/4): gd-2.0.35-11.el6.x86_64.rpm

| 142 kB 00:00

(2/4): nagios-3.5.1-1.el6.x86_64.rpm

| 1.2 MB 00:00 ...

(3/4): nagios-common-3.5.1-1.el6.x86_64.rpm

| 17 kB 00:00

(4/4): nagios-plugins-1.4.16-10.el6.x86_64.rpm

| 200 kB 00:00 ...

==========

Total

961 kB/s | 1.5 MB 00:01

Running rpm_check_debug

Running Transaction Test

Transaction Test Succeeded

Running Transaction

Installing : nagios-common-3.5.1-1.el6.x86_64

1/4

Installing : gd-2.0.35-11.el6.x86_64

2/4

Installing : nagios-3.5.1-1.el6.x86_64

3/4

Installing : nagios-plugins-1.4.16-10.el6.x86_64

4/4

Verifying : nagios-3.5.1-1.el6.x86_64

1/4

Verifying : gd-2.0.35-11.el6.x86_64

2/4

Verifying : nagios-common-3.5.1-1.el6.x86_64

3/4

Verifying : nagios-plugins-1.4.16-10.el6.x86_64

4/4

Installed:

nagios.x86_64 0:3.5.1-1.el6 nagios-plugins.x86_64 0:1.4.16-10.el6

Dependency Installed:

gd.x86_64 0:2.0.35-11.el6 nagios-common.x86_64 0:3.5.1-1.el6

Complete!

PRE-REQUISITOS:

yum install httpd php gcc glibc glibc-common gd gd-devel net-snmp-utils net-snmp openssl openssl-devel mod_ssl

Dependencies Resolved

==========

Package Arch Version Repository Size

==========

Installing:

gd-devel x86_64 2.0.35-11.el6 base
78 k
mod_ssl x86_64 1:2.2.15-30.el6.centos
updates 91 k
Updating:
glibc x86_64 2.12-1.132.el6_5.2
updates 3.8 M
glibc-common x86_64 2.12-1.132.el6_5.2
updates 14 M
httpd x86_64 2.2.15-30.el6.centos
updates 821 k
net-snmp x86_64 1:5.5-49.el6_5.1
updates 306 k
net-snmp-utils x86_64 1:5.5-49.el6_5.1
updates 174 k
openssl x86_64 1.0.1e-16.el6_5.14
updates 1.5 M
openssl-devel x86_64 1.0.1e-16.el6_5.14
updates 1.2 M
Updating for dependencies:
glibc i686 2.12-1.132.el6_5.2
updates 4.3 M
glibc-devel x86_64 2.12-1.132.el6_5.2
updates 978 k
glibc-headers x86_64 2.12-1.132.el6_5.2
updates 608 k
httpd-tools x86_64 2.2.15-30.el6.centos
updates 73 k
net-snmp-libs x86_64 1:5.5-49.el6_5.1
updates 1.5 M
nscd x86_64 2.12-1.132.el6_5.2

updates 219 k

Transaction Summary

==========

Install 2 Package(s)

Upgrade 13 Package(s)

Comprobamos que en la instalación a través de yum se resuelven también las dependencias.

Los ficheros de configuración de Nagios se encuentran almacenados en varios directorios y todos tienen extensión ".cfg" (ficheros de texto plano). Tras la instalación la estructura del directorio "/etc/nagios", debería contener estos archivos:

[miservidor-virtual]# *ls -lR*

.:

total 72

-rw-rw-r-- 1 root root 11658 Aug 31 2013 cgi.cfg

drwxr-x--- 2 root nagios 4096 Aug 31 2013 conf.d

-rw-rw-r-- 1 root root 44533 Aug 31 2013 nagios.cfg

drwxr-x--- 2 root nagios 4096 Aug 29 17:22 objects

-rw-r----- 1 root apache 27 Jun 16 13:24 passwd

drwxr-x--- 2 root nagios 4096 Jun 16 12:54 private

./conf.d:

total 0

./objects:

total 48

-rw-rw-r-- 1 root root 7704 Aug 31 2013 commands.cfg

-rw-rw-r-- 1 root root 2166 Aug 31 2013 contacts.cfg

-rw-rw-r-- 1 root root 5629 Jun 16 13:31 localhost.cfg
-rw-rw-r-- 1 root root 3124 Aug 31 2013 printer.cfg
-rw-rw-r-- 1 root root 3293 Aug 31 2013 switch.cfg
-rw-rw-r-- 1 root root 11158 Aug 31 2013 templates.cfg
-rw-rw-r-- 1 root root 3208 Aug 31 2013 timeperiods.cfg
-rw-rw-r-- 1 root root 4019 Aug 31 2013 windows.cfg

./private:
total 4
-rw-r----- 1 root nagios 1340 Aug 31 2013 resource.cfg

Donde los principales directorios y archivos son:

./nagios

- cgi.cfg:

  En este fichero se indica a los cgi's dónde deben encontrar los principales datos de la configuración de Nagios para su entorno web, como es la ubicación de ficheros HTML, usuarios autorizados para acceder a ciertos apartados de configuración vía web, tipos de gráficos, etc.

- nagios.cfg:

  Es el fichero más importante en la configuración de Nagios. En él se indicarán los ficheros que contienen la información de los objetos que debe monitorizar, dónde y cómo almacenar la información, cómo gestionar los ficheros de logs, así como, parámetros de configuración de la aplicación en sí.

./nagios/private

- resource.cfg:

  En él se indicarán los directorios donde se almacenan los recursos que va a utilizar Nagios durante su ejecución, por ejemplo, dónde se encuentran los plugins o los eventos que debe lanzar para solucionar algún problema. Se identificarán como “$USERn$”.

./nagios/objects

- commands.cfg:
  Es el fichero que reúne la forma en la que funcionarán los plugins. Si creáramos un plugin nuevo, deberíamos de declarar aquí cómo debería usarse: parámetros, ubicación, argumentos que necesita recibir, etc. También se indicará cómo configurar los mensajes de envío por SMS o email.

Los demás vienen por defecto y su nombre hace referencia a las definiciones que contienen. El nombre que se les proporciona a los ficheros de configuración debe ser el mismo al que se le haga referencia en el fichero “nagios.cfg”. En este TFG, los nombraremos de forma básica e intuitivos para poder comprender mejor la relación entre ellos:

- localhost.cfg:
  Contiene la información de los parámetros del propio servidor de Nagios que deseamos sean monitorizados. Es el único host con servicios predefinido.
- hosts.cfg:
  Definición de los hosts, servidores o dispositivos que se van a monitorizar
- services.cfg:

Definición de los servicios que se desea monitorizar

- contacts.cfg:
  Definición de los contactos
- templates.cfg:
  Plantillas para la definición “base” o “tipo” de hosts/servicios
- timeperiods.cfg
  Definición de intervalos de tiempo para monitorizar objetos
- printer.cfg:
  Descripción a modo de ejemplo de definición de impresora a ser monitorizada
- switch.cfg:
  Al igual que para las impresoras, un ejemplo para los dispositivos tipo switch donde configurar todos los dispositivos de la electrónica de red

Además de estos ficheros, Nagios admite una personalización de nuestra configuración de forma sencilla. Sólo hay que crear nuestros propios archivos “*.cfg” con nombres intuitivos que reúnan los objetos que queremos monitorizar según sean de naturaleza similar o porque se encuentren todos reunidos en una ubicación. Una vez creado, sólo habrá que indicarle a “nagios.cfg” el nombre y su ubicación.

Algunos ejemplos pueden ser:

- virtual-linux.cfg:
  Servidores virtuales con sistema operativo Linux

- sede-almeria.cfg:
  Objetos a monitorizar en la sede de Almería

- sensores.cfg:
  Definición de todos los sensores: consumo eléctrico, temperatura, etc

Adicionalmente, y por una cuestión meramente organizativa, se pueden crear agrupaciones con los hosts, servicios y contactos, de tal forma que añadiríamos los siguientes ficheros de configuración, cuyo contenido veremos en este apartado, más adelante y se comprenderá mejor su finalidad:

- hostgroups.cfg:
  Definición de grupos de hosts que se formarán con los hosts definidos en el fichero anterior

- servicegroups.cfg
  Definición de los grupos de servicios

- contactgroups.cfg:
  Definición de los grupos de contactos

Otros ficheros que suelen usarse en la comunidad Nagios son:

- escalations.cfg: se trata de establecer una escala para los avisos. Si una alerta sobre un servidor o servicio no es resuelta tras haber enviado un número de alertas, se empezarán a enviar a otros responsables, ya sean jefes o personal más cualificado.

- miscommands.cfg: es el fichero donde se suele configurar la información de las notificaciones por email o SMS. También puede definirse dentro de “commands.cfg”.

- dependencies.cfg: se pueden establecer relaciones entre servicios o hosts, que generen notificaciones u otros chequeos dependiendo del estado de los mismos.

- .htaccess / .htpasswd.users: se utiliza para limitar el acceso de usuarios al interfaz web. Para mejorar la seguridad, en el caso de implementación de TASA, se realizará a través de LDAP. Por lo que no se hará uso de estos ficheros.

Indicar que la distribución de las definiciones en distintos ficheros es opcional y se hace por jerarquizar la configuración, pero podemos simplificar la configuración y reunir en un sólo fichero el contenido de varios, siempre que respetemos la sencilla sintaxis de Nagios. Por ejemplo, en un sólo fichero hosts.cfg, podemos incluir también la configuración de los hostsgroups.cfg y hacer lo mismo para los servicios, etc. La flexibilidad y simplicidad para la configuración que presenta esta herramienta es uno de sus mayores valores.

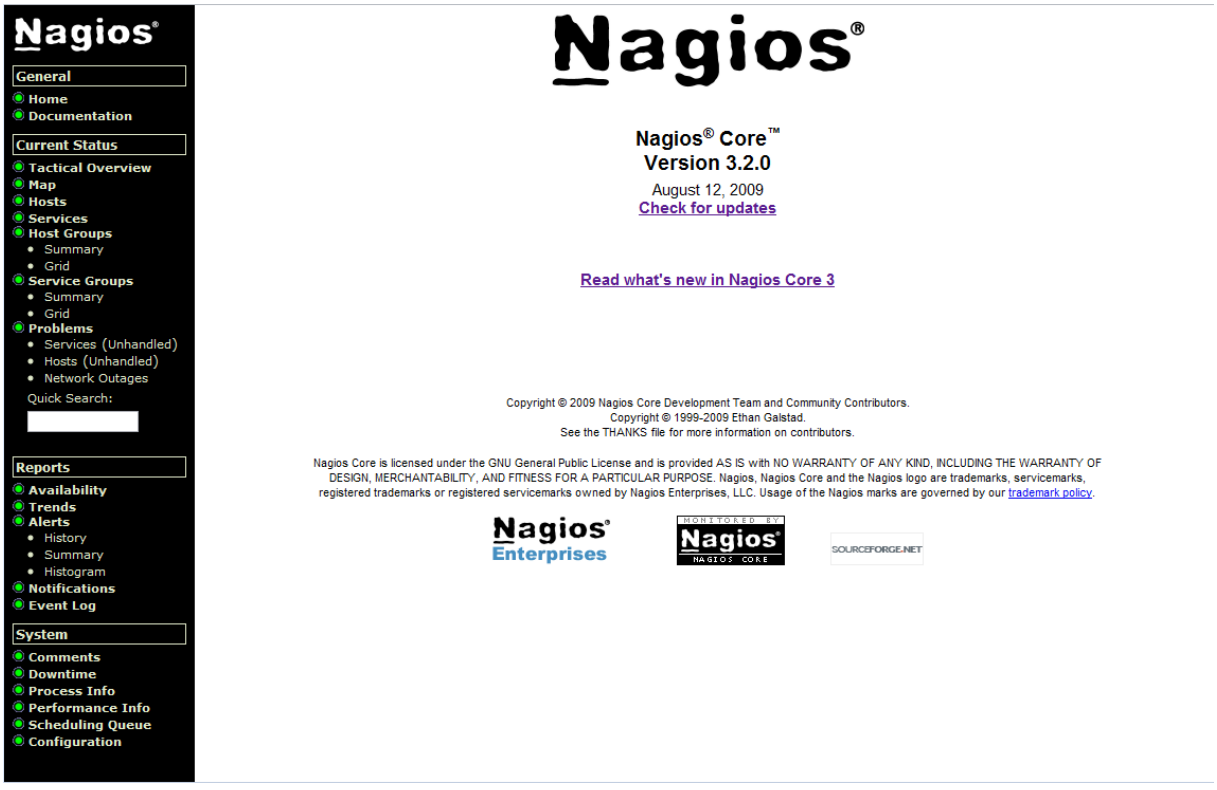


Figura 3.2. Interfaz gráfico de Nagios (Fuente: http://es.wikipedia.org)

Durante todo el proceso de instalación, Nagios dejará en /var/log/nagios/nagios.log, información de problemas que pudieran surgir para poder resolverlos.

Si todo ha ido bien hasta ahora, podríamos acceder al interfaz gráfico de nuestro Nagios como aparece en la **Figura 3.2,** aunque no hayamos definido nada aún.

3.2.2. Nagios en el cliente

En este TFG, detallaremos la instalación de dos tipos de clientes según el sistema operativo, ya sea para Linux o para Windows. Si algún dispositivo no tiene uno de estos sistemas operativos, se podrán utilizar otros protocolos para su monitorización como SNMP.

Adicionalmente, hay que tener en cuenta que Nagios no vigila sólo los componentes que pertenezcan a su rango de direccionamiento IP. Para ello, deberá configurarse la visibilidad entre el servidor principal de Nagios y el resto de todos los componentes. En caso de existir un cortafuegos, deberían definirse las reglas necesarias para permitir al servidor principal llegar al resto de objetos. Por lo que una vez instalado los clientes, se deberá comprobar la visibilidad entre servidor-cliente.

3.2.2.1. Cliente Nagios para Linux

El cliente Nagios para Linux es NRPE (Nagios Remote Plugin Executor). Es una aplicación que gestiona las peticiones que Nagios realiza sobre el estado de los parámetros que se han decidido monitorizar en cada sistema. En la **Figura 3.3**, se muestra cómo Nagios realiza una petición sobre un equipo Linux con el cliente instalado.

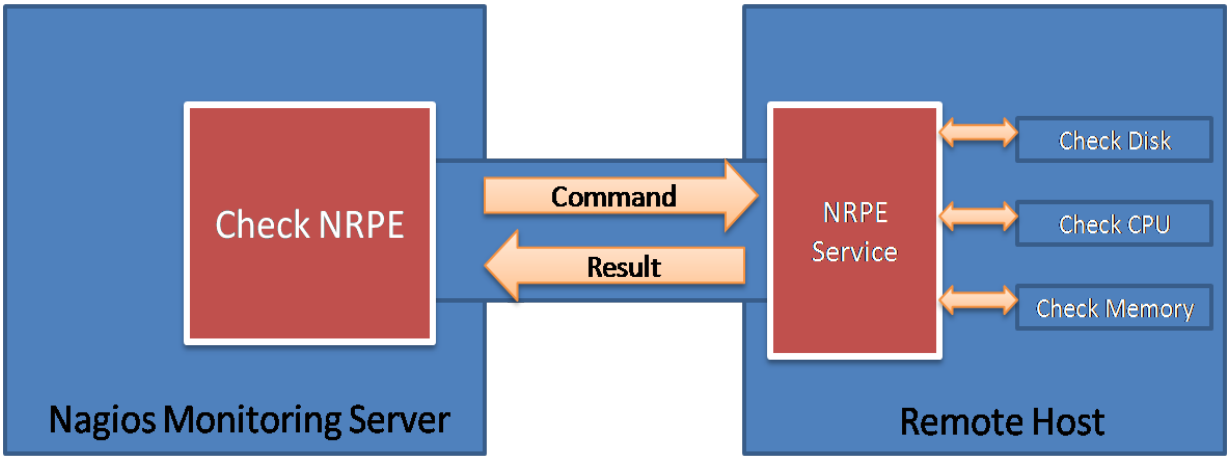


Figura 3.3. Interacción de Nagios con el cliente NRPE de sistemas Linux
(Fuente: http://tecadmin.net)

Para ello deberemos acceder al servidor donde queremos instalar el cliente y realizar los siguientes pasos: primero configurando los ficheros que permitirán la interacción con Nagios del propio sistema y después el del propio cliente NRPE para indicarle quién y qué va a monitorizarle:

- Instalación del cliente y los plugins
  * *yum install nagios-nrpe nagios-plugins*

- Cambiar los permisos y propietario para el fichero de configuración

  ** chown nagios:nagios /etc/nagios/*cfg*

- Editar /etc/hosts.allow para permitir que el servidor donde hemos instalado Nagios, pueda tener acceso

  ** vim /etc/hosts.allow y añadir 192.168.10.3*

- Editar /etc/services para que el puerto del servicio Nagios, 5666, esté

preparado para atender peticiones de este tipo

** vim /etc/services y añadir el puerto: "nrpe 5666/tcp # Nrpe Nagios"*

- Ahora, sí que se debe configurar el cliente de Nagios, NRPE. Para ello, tras editar su principal fichero de configuración, nrpe.cfg, deberemos darle permisos a la dirección IP del serviodor, para que pueda realizar peticiones a través del puerto indicado anteriormente, 5666. (Podremos añadir tantos servidores Nagios como deseemos que quieran monitorizar este sistema, separando cada dirección IP por comas)

  ** vim /etc/nagios/nrpe.cfg y modificar "allowed_hosts=192.168.10.3"*

- En este mismo fichero, “nrpe.cfg”, casi al final, se encuentran los servicios que vamos a permitir se “consulten” por parte del servidor principal de Nagios. Por ejemplo, en caso de la carga del sistema, deberíamos añadir una línea parecida a esta:

  ** command[check_load]=/usr/lib64/nagios/plugins/check_load -w 15,10,7 -c 30,25,20*

  que describiremos más adelante, junto con el resto de funcionamiento de los principales plugins, durante la configuración del sistema completo.

- Ya solo nos queda arrancar el servicio y añadirlo al arranque del sistema para cuando sea reiniciado.

  ** service nrpe start*
  ** chkconfig --level 345 nrpe on*

3.2.2.2. Cliente Nagios para Windows

El cliente Nagios para Windows es NSClient++.

Para el caso de monitorización de equipos windows, la instalación del cliente es similar.

Descargamos desde su página web (http://www.nsclient.org/download/) la opción correspondiente a nuestra versión de Windows y ejecutamos el asistente para su configuración.

Los parámetros a configurar, serán los mismos que en el caso de cliente Linux: el puerto 5666 e IP del servidor Nagios, 192.168.10.3.

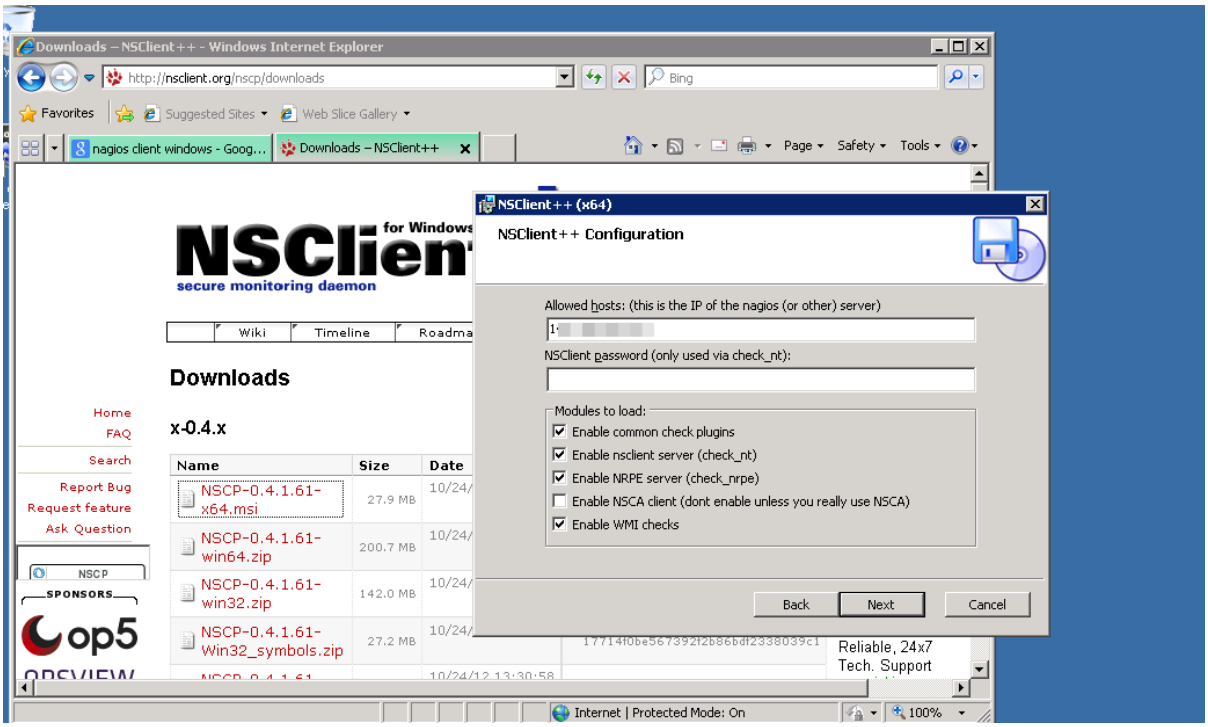


Figura 3.4. Imagen de instalación (Fuente: http://linuxdrops.com/)

En la **Figura 3.4** se muestra una imagen del asistente para su instalación.

Por último, sólo debemos dejar arrancado el servicio en el cliente Windows.

3.2.3. Plugins

Un plugin es un sencillo programa software (el cuál puede estar generado en distintos lenguajes de programación como Perl, Python, C,...) cuya función es comprobar el estado de lo que queremos monitorizar, dentro de unos umbrales de error, que serán especificados en la invocación del plugin. Existe una amplia comunidad implicada en Nagios, en la que continuamente se aportan nuevos plugins o versiones de los existentes, para las nuevas necesidades que van surgiendo a medida que evolucionan los servicios y los dispositivos objetos de monitorización. Podemos comprobarlo en http://www.nagiosexchange.org.

Tanto en el caso del servidor como en los clientes, durante la instalación, se instalan los plugins para que el servidor pueda controlar los componentes, aplicaciones o servicios de cada sistema.

La información se intercambiará entre los sistemas a través de diferentes protocolos de comunicación, como SNMP, o de la respuesta de los clientes instalados, NRPE y NsClient+.

De los plugins más comunes e importantes, nos encontramos con:

check_ping: el más sencillo y a la vez más útil. Todo dispositivo que posea una dirección IP, podrá ser monitorizado.

check_tcp: Para confirmar que un puerto está escuchando correctamente en un servidor destino.

check_http: comprueba que una página web está visible y con algunas opciones,

hasta saber los días que le quedan al certificado https

check_dns: Si el servidor de nombres está resolviendo bien

check_load: Conocer la carga de un sistema

check_mem: El estado de la memoria, cuánto le queda libre

check_disk : Espacio en disco disponible e inodos. Parametrizándolo, conoceremos la información de cualquier disco del sistema (físico o virtual)

check_nrpe: para los sistemas Linux que tengan el cliente instalado

check_nt: para los sistemas Windows que tengan el cliente instalado

En el **Anexo 1**, se listan los plugins que proporciona Nagios por defecto.

Añadir, que los plugins pueden ser ejecutados desde la línea de comando del propio servidor, para comprobar que los parámetros y umbrales definidos en su llamada son correctos, antes de incluirlos en la configuración. Veamos un ejemplo:

> ***[miservidor-virtual]#*** *`/usr/lib64/nagios/plugins/check_tcp -H IPADDRESS -p 1433 -w 5 -c 10`*
> *`TCP OK - 0.001 second response time on port 1433|`*
> *`time=0.000579s;5.000000;10.000000;0.000000;10.000000`*

Comprobamos el estado de un puerto, en este caso el 1433, mediante el plugin check_tcp, sobre el servidor con dirección IP, IPADDRESS, responde correctamente para tiempos inferiores a 5 segundos (si es mayor indicarían que su estado es WARNING) y menores de 10 segundos (si supera este umbral, marcará el estado como CRITICAL).

Del mismo modo se puede realizar para cualquier tipo de plugin, incluso para servidores con clientes instalados:

> ***[miservidor-virtual]#*** *usr/lib64/nagios/plugins/check_nrpe -H IPADDRESS -c copias_base_datos*
> *FILE_AGE OK: /copias_bbdd/fichero_dump is 86337 seconds old and 104736220690 bytes*

donde a través del cliente NRPE, con dirección IP, IPADDRESS, ejecutaríamos el plugin tal y como se definió en su fichero “nrpe.cfg” del cliente. En este caso comprobando la fecha y el tamaño del fichero creado tras la copia de seguridad realizada.

Una vez comprobados, ya podremos añadirlos a los ficheros de configuración, como veremos a partir del próximo apartado.

3.3. Configurando Nagios

Los ficheros básicos de Nagios anteriormente nombrados, son los que reúnen toda la información acerca de qué, cómo, cuándo queremos monitorizar y a quién y cómo queremos notificar en caso de alerta.

A continuación, vamos a describir cómo se deben configurar y la relación entre ellos, para que Nagios interprete nuestras necesidades.

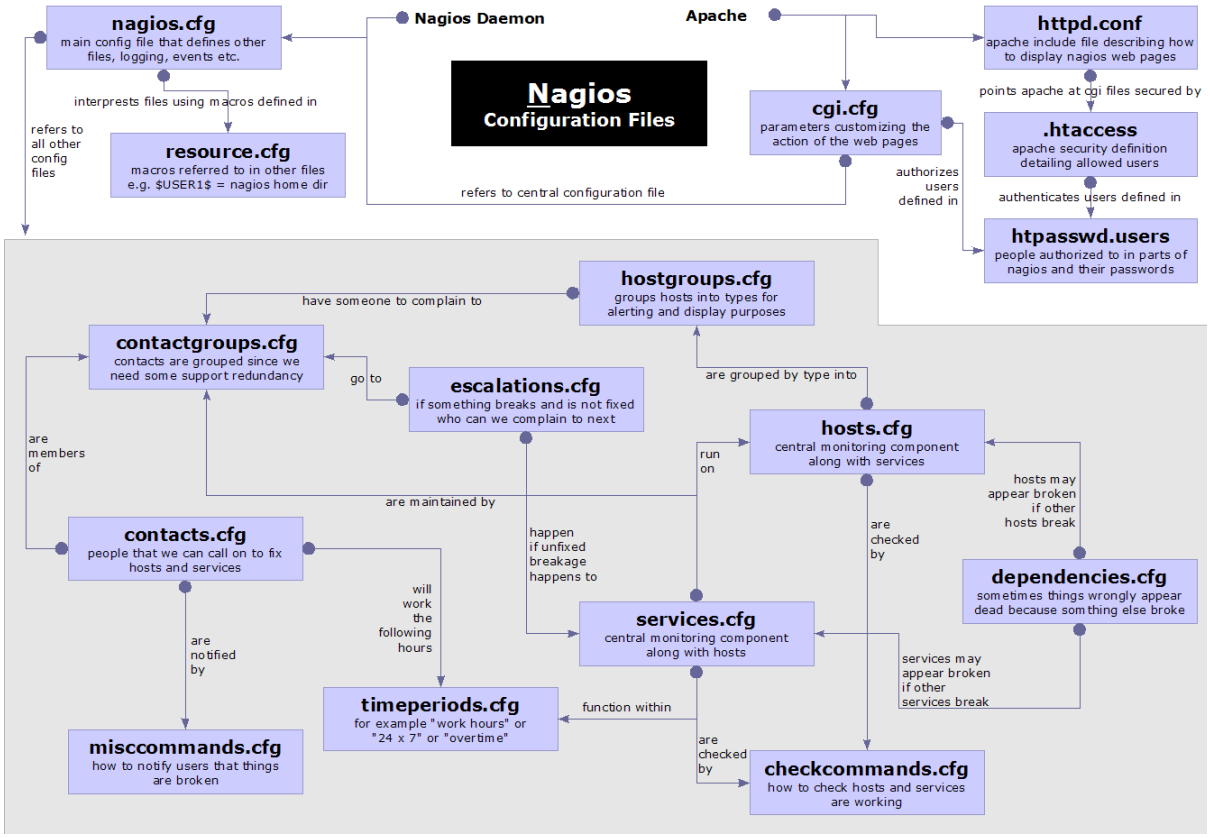


Figura 3.5. Ficheros básicos de Nagios (Fuente: http://www.the-tech-tutorial.com/)

Según vemos en la **Figura 3.5**, existe una relación entre los ficheros básicos de configuración que nos proporciona Nagios y que deberán ser actualizados con la información de nuestra red informática. Por la flexibilidad de Nagios, que ya hemos

mencionado anteriormente, estos archivos pueden renombrarse, añadir nuevos, agruparse, etc.

3.3.1. hosts.cfg

Empezamos por los “hosts”. Un host para Nagios es todo dispositivo con una dirección IP asignada al que puede acceder para monitorizar, como mínimo que está encendido, es decir, que responde correctamente a una señal de PING enviada. En un entorno práctico, un host puede ser: un servidor (físico o virtual, windows o linux), la principal línea de datos o de respaldo de una empresa, un sensor de temperatura o consumo eléctrico, un sistema de alimentación ininterrumpida (SAI), un switch, una impresora, un AP para wifi, etc,....

```
define host {
        use                     generic-host
        host_name               NombreHost
        alias                   AliasHost
        address                 192.168.10.5
        icon_image              base/linux40.gif
        statusmap_image         base/linux40.png
        parents                 NombreNodoPadre
        notes_url       $USER1$/cacti/cactiplug.php?ip=$HOSTADDRESS
}
```

Estos son los principales parámetros con los que se describe un host:

use: plantilla (se detallará más adelante) de la que “hereda” el resto de los valores que no se especifican en esta definición. En el caso de necesitar algún valor diferente definido en la plantilla, para un host en concreto, sólo hay que volver a definirlo aquí, y este valor prevalece, frente al de la plantilla.

host_name: nombre con el que se hará referencia al objeto, dentro de los ficheros de
configuración de Nagios

alias: nombre del host que visualizará el interfaz gráfico, al cual se le puede asignar una etiqueta más descriptiva, si no se indica, aparecerá el nombre asignado en "host_name"

address: es la dirección IP a la que Nagios debe acceder para monitorizar este objeto

icon_image: icono que deseamos se muestre al presentar el host en el interfaz web

statusmap_image: icono para identificarlo en el mapa de estado de Nagios

parents: en el caso de que así sea, indicamos de qué otro/s servidor/es depende. Básicamente, se empleará para establecer una jerarquía, que será muy útil para monitorizar y realizar la representación de los mapas del sistema. En algunos casos, se puede especificar más de un valor, según la relación existente entre ellos

notes_url: lo explicaremos más adelante, pero a través de este parámetro, proporcionamos un enlace a otra url que contiene información ampliada de este host.

3.3.2. services.cfg

¿Qué tipos de servicios puede monitorizar Nagios? Pues realmente casi cualquiera que deseemos: una página web, un servidor de nombres, un servidor de bases de datos, el tamaño de un sistema de archivos, si un puerto de un switch tiene tráfico, si un certificado web está a punto de caducar, número de usuarios conectados a un

servidor, etc y lo mejor de esta herramienta es, que si no existe el plugin al que el servicio pueda invocar para conocer el estado, siempre nos quedará la opción de desarrollarlo nosotros mismos y contribuir a la comunidad de Nagios, por si puede necesitarlo alguien con nuestras mismas necesidades.

La configuración de los servicios tampoco es demasiado complicada, pero hay que diferenciar entre varios conceptos: definición del servicio genérico, definición del comando que ejecuta el plugin y los datos de nuestro servicio a monitorizar según lo anteriormente establecido.

Una vez más, todo esto se ve más claro con un ejemplo. Como comentamos en el caso de los hosts, el servicio más sencillo a monitorizar será la respuesta al PING de un host, pues bien, para ello deberemos configurar en este fichero:

```
define service{
   use                    generic-service
   name                   ServicioPing
   service_description    DescServicio
   check_command          check_ping!500.0,30%!800.0,60%
   }
```

Al igual que ocurría en el caso de definición de los hosts, para los servicios también heredamos la mayoría de los valores de las plantillas definidas en “templates.cfg”, indicando aquí, los valores concretos para este servicio, como son:

use: plantilla de la que partimos (se detallará más adelante)

name: nombre del servicio

service_description: etiqueta con la que se visualizará el servicio en la web.

check_command: los datos de qué plugin queremos utilizar para este servicio y sus valores de parametrización. En este caso, este servicio monitoriza el PING, haciendo uso del plugin “check_ping” al que se le pasan los parámetros tras el signo de exclamación.

Además, indicaremos dentro de este mismo fichero, qué host es el que queremos monitorice el servicio de PING:

```
define service{
    use                 ServicioPing
    host_name           NombreHost1, NombreHost2
    }
```

Para que esto funcione, comprobaremos que en el fichero “commands.cfg” aparece la entrada para este pluging, que recibe los valores que acabamos de definir en nuestro servicio:

```
# 'check_ping' command definition
define command{
    command_name    check_ping
    command_line    $USER1$/check_ping -H $HOSTADDRESS$ -w
 $ARG1$ -c $ARG2$
    }
```

command_name: es el nombre del servicio, que debe coincidir con el usado en nuestra definición del servicio en el parámetro “check_command”

command_line: es el que se encarga de ejecutar el comando en sí. Busca el script “check_ping” en el directorio que habremos configurado como “$USER1$”. Lo ejecutará con los parámetros indicados, -w, para el umbral de aviso o Warning, $ARG1$, y -c para el crítico o Critical, $ARG2$, y para el host especificado,

$HOSTADDRESS$, es la dirección IP de NombreHost1 de nuestro ejemplo. Hay que tener en cuenta, tanto para este plugin, como para cualquier otro, que los umbrales pueden ser totalmente distintos dependiendo del sistema en el que lo ejecutemos y según el interés o criticidad que tenga. Este es otro factor que indica la versatilidad y flexibilidad de Nagios.

Este fichero “commands.cfg”, trae por defecto la definición de los parámetros y forma de invocar los plugins básicos de Nagios. En caso de añadir nosotros un nuevo plugin, deberemos, además de ubicarlo en el directorio correcto, definirlo en este fichero para que pueda atender las llamadas realizadas desde ficheros como “services.cfg”.

En el caso de la definición de servicios, debemos hacer una descripción más detallada de la definición de servicios para el caso de que interactúen con los clientes NRPE y NsClient+, anteriormente descritos.

La definición del servicio es básicamente la misma, mostramos un ejemplo y a continuación lo describiremos. Vamos a ver cómo se definirían los servicios que comprueban la carga de un sistema para el caso de NRPE (cliente Linux):

```
define service{
  use                    generic-service
  host_name              NombreHost1
  service_description    Carga Actual
  check_command          check_nrpe!check_carga
  }
```

En este caso, con el plugin check_nrpe, Nagios solicita información al servidor NombreHost1. Éste tiene el cliente NRPE instalado y en su fichero de configuración, “nrpe.cfg”, hay una entrada para comprobar la carga, “check_carga” con sus correspondientes parámetros, como vimos antes:

```
…….....
command[check_carga]=/usr/lib64/nagios/plugins/check_load -w 15,10,7 -c
30,25,20
…....
```

Como vimos antes en el caso del servicio de PING, la opción “-w” indicará el umbral de aviso para “Warning” y la “-c” para el caso “Critical”.

Y para el caso de NsClient+ (cliente Windows):

```
define service{
        use                     generic-service
        host_name               NombreHost1, NombreHost2
        service_description     CPU Load
        check_command           check_nt!CPULOAD!-l 5,80,90
        }
```

la sintaxis es igual, pero en este caso se le proporciona los valores en la misma consulta, sin tener que configurar nada en el cliente, como ocurría anteriormente.

Para terminar la configuración de servicios, deberemos comentar la definición de lanzamiento de eventos que puedan realizar una acción adicional una vez se produzca un cambio de estado:

```
define service{
…....
event_handler     reinicia_httpd!$SERVICESTATE$!$SERVICEATTEMPT$
event_handler_enabled   1
….....
}
```

Con ello conseguiríamos ejecutar el script “reinicia_httpd” cuando haya un error.

3.3.3. hostgroups.cfg

Otros ficheros de configuración son los que agrupan los hosts o servicios, como son “hostsgroups.cfg” y “servicegroups.cfg”. Este tipo de agrupamiento de servidores o servicios, nos ayudará en la simplificación de las configuraciones, y además, en el interfaz gráfico aparecerán agrupados según los definamos, con lo que conseguiremos un mejor entendimiento de nuestra organización y mejor visualización de las alertas.

Para el caso de servidores, las definiciones en “hostgroups.cfg” serían algo así:

```
define hostgroup{
        hostgroup_name      NombreGrupoHost
        alias               Grupo1 Servidores
        members             NombreHost1, NombreHost2
}
```

Con esta agrupación, partir de este momento, podemos monitorizar el mismo servicio para todo un grupo de servidores de forma sencilla, con definiciones de este tipo:

```
define service{
        use                         ServicioPing
        hostgroup_name              NombreGrupoHosts1, NombreGrupoHost2
        }
```

Con esta definición, controlamos el servicio del PING de todos los servidores

agrupados, no sólo en NombreGrupoHosts1, sino también en otro grupo denominado NombreGrupoHosts2. Pudiendo añadir todos los grupos que queramos separando los nombres por comas.

3.3.4 servicegroups.cfg

En este caso, se define un grupo de servicio, siguiendo con nuestro ejemplo, para el servicio de PING, la configuración sería así:

```
define servicegroup
{
        servicegroup_name       GrupoServicioPing
        members                 NombreHost1,PING, NombreHost2,PING
}
```

3.3.5. contacts.cfg

Llegado este momento, hemos configurado nuestro Nagios con los servidores, servicios y plugins. Falta la otra parte, no menos importante, para finalizar la configuración. Ahora debemos definir quienes son los responsables, o también llamados “contactos”, a los que se debe avisar en caso de que se produzca alguna alerta y definir en la vía en la que se debe hacer la notificación.

Dentro de este fichero se configurarán uno a uno todos los contactos (responsables) a los que se les deban enviar un aviso cuando se produzca una alerta en el sistema de monitorización. La declaración de cada uno de ellos debe ser así:

```
define contact
{
```

```
        contact_name        nagiosadmin1
        use                 generic-contact
        alias               Administrador NAGIOS
        email               nagiosadmin@midominio.es
        pager               +34666123456
    }
```

contact_name: es el nombre del contacto

use: indica la plantilla de la que hereda el resto de los parámetros

alias: etiqueta con la que aparecerá en el interfaz gráfico

email: dirección de correo electrónico a la que se le enviarán las notificaciones

pager: número de móvil al que enviar los SMS de alerta.

3.3.6. contactgroups.cfg

Siguiendo con la simplificación a la hora de configurar Nagios, con los contactos volveríamos a agruparlos en la medida de lo posible para hacer referencia a grupos de contactos que deban recibir las notificaciones. La declaración de los grupos sería de esta forma:

```
    define contactgroup{
        contactgroup_name       nagiosadmin
        alias                   Administradores NAGIOS
        members                 nagiosadmin1, nagiosadmin2
    }
```

contactgroup_name: es el nombre del grupo dentro de nuestra configuración

alias: como en ocasiones anteriores, es la etiqueta con la que se visualizará en el entorno web

members: son todos los “contact_name” definidos anteriormente, que deseemos agrupar en este caso.

3.3.7. timeperiods.cfg

Hasta el momento, hemos definido todo lo que queremos monitorizar, de qué forma y a quién debemos hacer llegar los avisos por alertas. Bueno, pues ahora debemos establecer los períodos de tiempo en los que deseamos que se realicen. Para ello, existe este fichero, donde se definen de la siguiente forma:

```
define timeperiod{
        timeperiod_name     24x7
        alias               24 Horas al Día, 7 Días a la semana
        sunday              00:00-24:00
        monday              00:00-24:00
        tuesday             00:00-24:00
        wednesday           00:00-24:00
        thursday            00:00-24:00
        friday              00:00-24:00
        saturday            00:00-24:00
        }

define timeperiod{
        timeperiod_name     smshoras
```

alias SMS Horas
sunday 08:00-24:00
monday 08:00-24:00
tuesday 08:00-24:00
wednesday 08:00-24:00
thursday 08:00-24:00
friday 08:00-24:00
saturday 08:00-24:00
}

timeperiod_name: nombre que se le proporciona a este período de tiempo

alias: nombre con el que se visualiza en el entorno gráficos

monday – sunday: por cada día de la semana, se especifica un intervalo de horas. De 00:00 a 24:00, si queremos todo el día, o por ejemplo, para el envío de SMS, de 08:00 a 24:00. Dependiendo de la criticidad del sistema a monitorizar se podrá ampliar o reducir el intervalo de tiempo.

3.3.8. templates.cfg

Nagios presenta un gran número de parámetros para la configuración de los objetos a monitorizar. Para la definición de los mismos, lo más recomendable es crear una plantilla, con todos los parámetros comunes a un gran número de objetos / servicios / contactos / períodos de tiempo y después, en los ficheros como los que acabamos de describir, (hosts.cfg, services.cfg, contacts.cfg,..) crear una entrada para cada uno de ellos, con sus valores propios, indicando que el resto de valores, los tomará de la plantilla indicada. Dependiendo del grupo de elementos a monitorizar se definirán valores distintos para cada opción.

Así, para los dispositivos más sencillos y poco críticos, podremos establecer que

sólo nos alerte cuando los cambios de estado se produzcan de CRITICAL a OK y viceversa. En cambio para los más decisivos, le indicaremos que cada vez que haya cualquier cambio de estado nos envíe una notificación.

Otros parámetros que podemos indicar será la periodicidad con la que queremos que se efectúen los chequeos, grupos o contactos a los que enviar notificaciones, tiempo que debe transcurrir entre cada envío de alerta, etc.

Si al definir un servidor o servicio, no queremos que todos los parámetros coincidan exactamente con los de la plantilla, bastará con añadir a su definición el nuevo valor para ese parámetro en concreto.

Ejemplos de plantillas y descripción de los parámetros más importantes pueden verse en el **Anexo 2**.

Hasta ahora hemos sólo hemos visto cómo configurar Nagios, ha llegado la hora de ver cómo quedaría todo en el interfaz gráfico. Para ello, Nagios nos proporciona un comando de depuración algo básico, pero bastante útil. Con la opción "-v" e indicando el fichero de configuración, nos mostrará si existe algún error de sintaxis o de relación, mostrando en qué fichero está el error y si es sólo de aviso (Warning) o crítico (Critical).

```
[miservidor-virtual]# nagios -v nagios.cfg

Nagios Core 3.5.1
Copyright (c) 2009-2011 Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 08-30-2013
License: GPL
```

Website: http://www.nagios.org
Reading configuration data...
Read main config file okay...
Processing object config file '/etc/nagios/objects/commands.cfg'...
Processing object config file '/etc/nagios/objects/contacts.cfg'...
Processing object config file '/etc/nagios/objects/timeperiods.cfg'...
Processing object config file '/etc/nagios/objects/templates.cfg'...
Processing object config file '/etc/nagios/objects/localhost.cfg'...
Processing object config directory '/etc/nagios/conf.d'...
Read object config files okay...

Running pre-flight check on configuration data...

Checking services...
Checked 9 services.
Checking hosts...
Checked 2 hosts.
Checking host groups...
Checked 1 host groups.
Checking service groups...
Checked 0 service groups.
Checking contacts...
Checked 1 contacts.
Checking contact groups...
Checked 1 contact groups.
Checking service escalations...
Checked 0 service escalations.
Checking service dependencies...
Checked 0 service dependencies.
Checking host escalations...
Checked 0 host escalations.

Checking host dependencies...

Checked 0 host dependencies.

Checking commands...

Checked 24 commands.

Checking time periods...

Checked 5 time periods.

Checking for circular paths between hosts...

Checking for circular host and service dependencies...

Checking global event handlers...

Checking obsessive compulsive processor commands...

Checking misc settings...

Total Warnings: 0

Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check

si el resultado muestra que no existe ningún error, debemos reiniciar el servicio:

[miservidor-virtual]# *service nagios restart*

y ya podremos acceder a nuestra configuración en el interfaz gráfico.

3.4. Interfaz web

Una vez realizada toda la instalación y configuración, deberemos confirmar que todo ha quedado como deseábamos a través de nuestro interfaz web. A los ficheros de configuración ya no debemos acceder, a no ser que debamos añadir o corregir algo puntualmente. A partir de ahora, accederemos a toda la información de nuestra red informática monitorizada de forma gráfica, mediante el acceso web, o bien, por complementos de navegadores o aplicaciones móviles (estos últimos los veremos en el próximo capítulo).

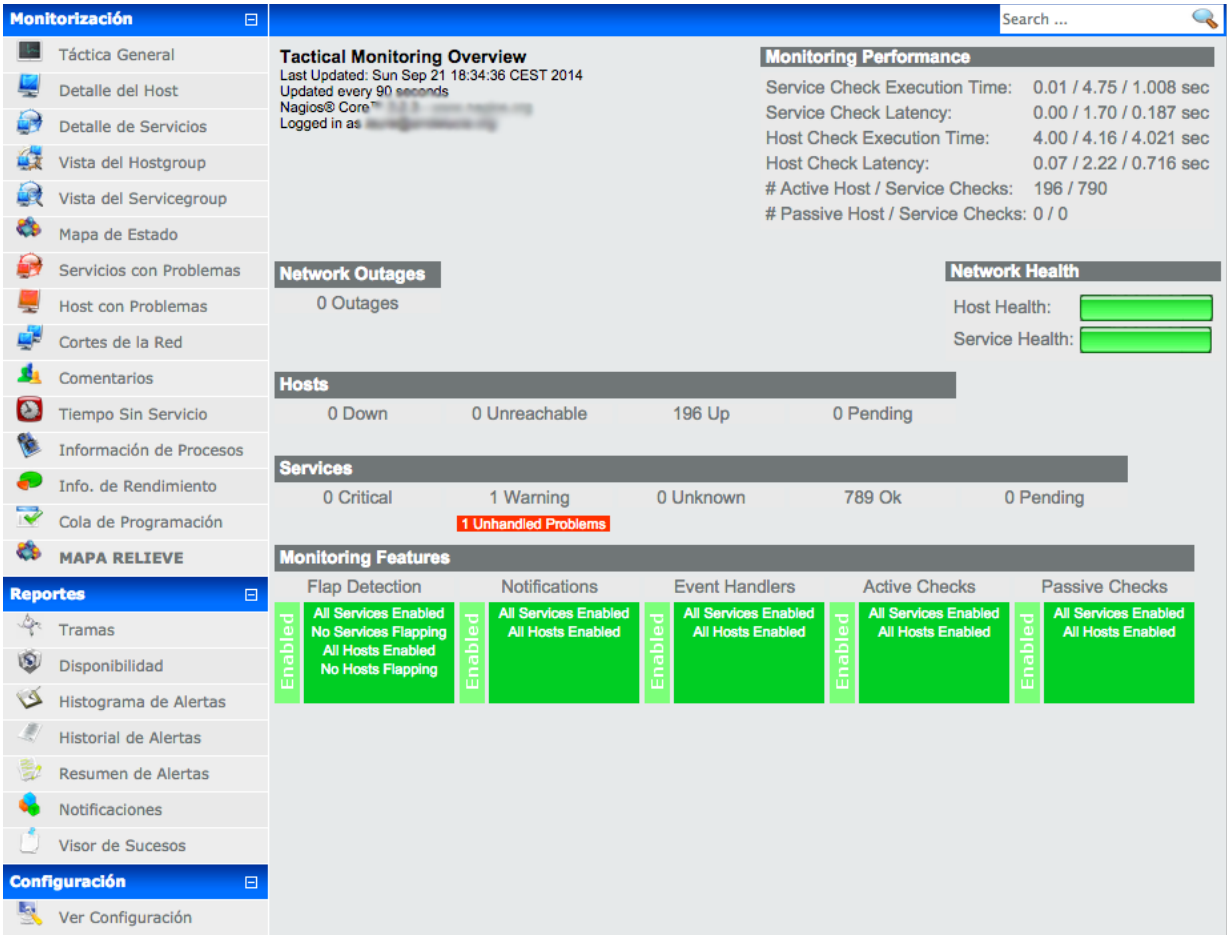


Figura 3.6. Interfaz resumen de estado

Como hemos comentado anteriormente, Nagios es una herramienta con una gran

comunidad que a lo largo de los años la ha ido evolucionando y actualizando. Una de las aportaciones han sido los distintos temas o skins para el interfaz de usuario de la aplicación. De los disponibles, el que consigue un diseño más limpio y menos llamativo, al tener una gama de colores más suave es el de Arana (creado por Eduardo Arana). Por lo que nuestra página inicial debería parecerse bastante a esta imagen de la **Figura 3.6**.

Vayamos a describir las principales pantallas de Nagios. El interfaz, a primera vista, como podemos comprobar es básico y sencillo. Consta de una columna izquierda que nos presentará la opciones de los diferentes contenidos que podremos seleccionar y una pantalla central con el contenido de las mismas.

Las etiquetas son bastante intuitivas y podremos predecir la información que nos va a presentar cada una de ellas. Debemos tener en cuenta, que según hayamos definido en el fichero “cgi.cfg”, el usuario logado podrá acceder a todas o sólo a algunas de las opciones, dependiendo del nivel de responsabilidad que tenga.

Por ejemplo, en la **Figura 3.6**, se muestra la pantalla inicial, en la que se recoge a modo de resumen el estado de la red monitorizada que nos proporciona Nagios por defecto. En nuestro caso, recordar que la configuración inicial sólo define a “localhost”, es decir, al servidor de Nagios en sí, por lo que nos mostrará un sólo servidor (host) y siete servicios (services) vigilados, de los que, en este caso, uno de ellos presenta un estado de “WARNING” y el resto “OK”. Una vez hayamos añadido a la configuración todos nuestros objetos, el interfaz mostrará toda la información.

La columna de opciones se encuentra dividida en tres secciones: Monitorización, Reportes y Configuración.

- Monitorización: En esta sección accederemos a la información de todos los objetos de diversas formas, tanto individualmente como en forma de red o de grupo, para tener todas las perspectivas posibles y comprender mejor cómo

está configurada la red. Esto nos simplificará en caso de error, la toma de decisiones para corregirlo. De este modo, podremos ver:

- Resumen o “Táctica General”: en esta página se presenta un resumen general del estado de toda la red monitorizada, indicando según el estado, el número de elementos:

    - Servidores: Down – Unreachable – Up – Pending. Es decir, cuántos están apagados, inalcanzables (por jerarquía hay algún dispositivo que no le permite llegar a él para conocer su estado), funcionando y, por último, pendiente, porque aún no ha obtenido la información para definir su estado.

    - Servicios: Critical – Warning – Unkown – Ok – Pending: Se encuentra en estado de alerta crítico, en nivel de aviso, desconocido (que no puede deducirlo), correcto o pendiente (igual que para los servidores)

    - Por último, también nos presenta a modo de resumen, cómo se encuentra el proceso de monitorización: *Flap Detection*, quiere decir que el objeto ha estado cambiando de estado demasiadas veces en un corto período de tiempo y, por tanto, que lo pone en vigilancia hasta que esté estable. *Notifications*, por si hemos decidido que debemos silenciar las notificaciones de algún objeto (lo están reparando o es un corte de servicio controlado), *Event Handlers*, si se ha desactivado la opción de lanzar eventos cuando ocurra alguna alerta, *Active – Passive Checks*, si todos los objetos tienen activados los chequeos, bien los que hace Nagios directamente o de los que espera información por tener clientes remotos.

- Listado de cada dispositivo y de cada servicio

- Agrupados según hayamos definido en hostgroups y servicegroups, respectivamente

- Ver un mapa de estado en el que se posicionen todos los objetos y la jerarquía que hemos establecidos entre ellos

- Listado sólo de los hosts y servicios que tienen en este momento algún problema (su estado es diferente de “OK”)

- Cortes que se hayan producido en la red, caídas de líneas de datos

- Comentarios que se van añadiendo para hosts y services a tener en cuenta antes/durante/después de las incidencias

- Tiempos en los que por algún motivo ha estado sin monitorizarse, por un reinicio del sistema, porque se va a realizar mantenimiento en el CPD, etc, se debe documentar para tener en cuenta al analizar los datos de estado de los hosts y servicios.

- Los superusuarios también podrán cambiar parámetros de la configuración desde la “Información de Procesos”, donde se podrá reiniciar el servicio de Nagios, apagarlo, deshabilitar todas las notificaciones, detener todos los chequeos, etc.

- Tener información acerca del rendimiento que tiene actualmente

- Comprobar el estado de la “Cola de Programación”, orden en el que se

van a realizar los próximos chequeos.

- Reportes: En este apartado, podremos generar informes del estado de toda la red según nuestras necesidades de información. Proporciona multitud de combinaciones para seleccionar los objetos, el período y demás parámetros que queremos recopilar y crear los resúmenes que necesitemos. Además, podemos ver las últimas notificaciones enviadas y un "Visor de Sucesos", con los últimos cambios de estado que se han producido en los componentes de la red.

- Configuración: Sólo disponible para el personal responsable de Nagios y de la red, mostrará toda la configuración que hayamos plasmado en los ficheros de configuración, al igual que antes, nos proporciona formularios para agrupar la información de la que realmente queremos comprobar la información.

A continuación, se muestran algunas capturas de pantalla que acabamos de describir (**Figuras 3.7** y **3.8**):

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	09-21-2014 19:30:15	19d 22h 12m 4s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	09-21-2014 19:30:12	19d 22h 10m 57s	1/4	USERS OK - 0 users currently logged in
	HTTP	WARNING	09-21-2014 19:30:55	19d 22h 9m 51s	4/4	HTTP WARNING: HTTP/1.1 401 Authorization Required - 724 bytes in 0.001 second response time
	PING	OK	09-21-2014 19:31:38	19d 22h 8m 44s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	09-21-2014 19:27:21	19d 22h 12m 37s	1/4	DISK OK - free space: / 1174 MB (38% inode=9%):
	SSH	OK	09-21-2014 19:28:04	19d 22h 11m 31s	1/4	SSH OK - OpenSSH_5.3 (protocol 2.0)
	Total Processes	OK	09-21-2014 19:28:47	19d 22h 9m 17s	1/4	PROCS OK: 6 processes with STATE = RSZDT

Results 1 - 7 of 7 Matching Services

Figura 3.7. Estado de los servicios definidos para localhost

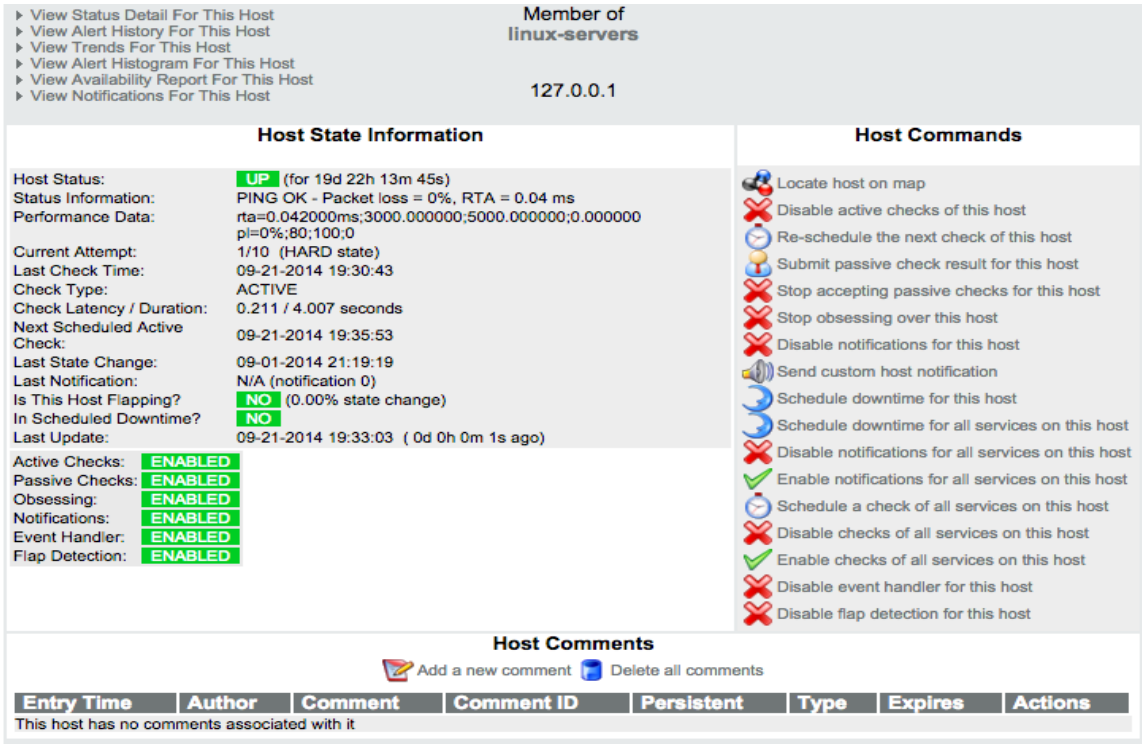


Figura 3.8. Estado de "localhost" y posibles acciones

En estos ejemplos, podemos ya intuir el potencial que va a tener Nagios una vez configuremos nuestra red. Para un solo servidor, podemos ver toda la información de su estado, configuración e incluso realizar acciones directamente sobre él desde el interfaz web, como por ejemplo, silenciar las notificaciones, porque es un error conocido y ya estamos en vías de solucionar el problema.

3.5. Otras opciones para comprobar estados y alertas

Como ya se ha comentado a lo largo de este TFG, la forma de interactuar con Nagios será vía Web en la que podremos comprobar el estado de todos nuestros objetos monitorizados y recibiendo avisos de alertas vía email o SMS. Pues bien, además existen otras opciones muy útiles como son los complementos gráficos para los navegadores web y las aplicaciones para móviles o tabletas. Suelen mostrar un resumen del estado global de incidencias en la posición de la pantalla que le indiquemos (por ejemplo, en la barra de estado). Éstas interactúan mostrando el cambio de estado de algún dispositivo en ventanas o pop-ups, o también, emitiendo sonidos por si no hay nadie frente al monitor en ese momento.

3.5.1. Complementos para navegadores web

También conocidos como “addons”, se instalan en los navegadores web que usamos habitualmente. Tras ser configurado con los datos de nuestro servidor de monitorización, podremos ser avisados del estado de todo nuestro sistema, de la forma que indiquemos y nos proporcione este complemento: señales visuales en la barra de estado, aparición de nuevas ventanas mostrando la notificación, emisión de algún tipo de sonido, etc.

El complemento más empleado hoy día por su simplicidad para ser configurado y eficiencia es “Nagios CheCker”. En esta capturas a modo de ejemplo, se muestran algunas configuraciones e iconos de estado de Nagios, para Chrome (**Figura 3.9**) y Firefox Mozilla (**Figuras 3.10**).

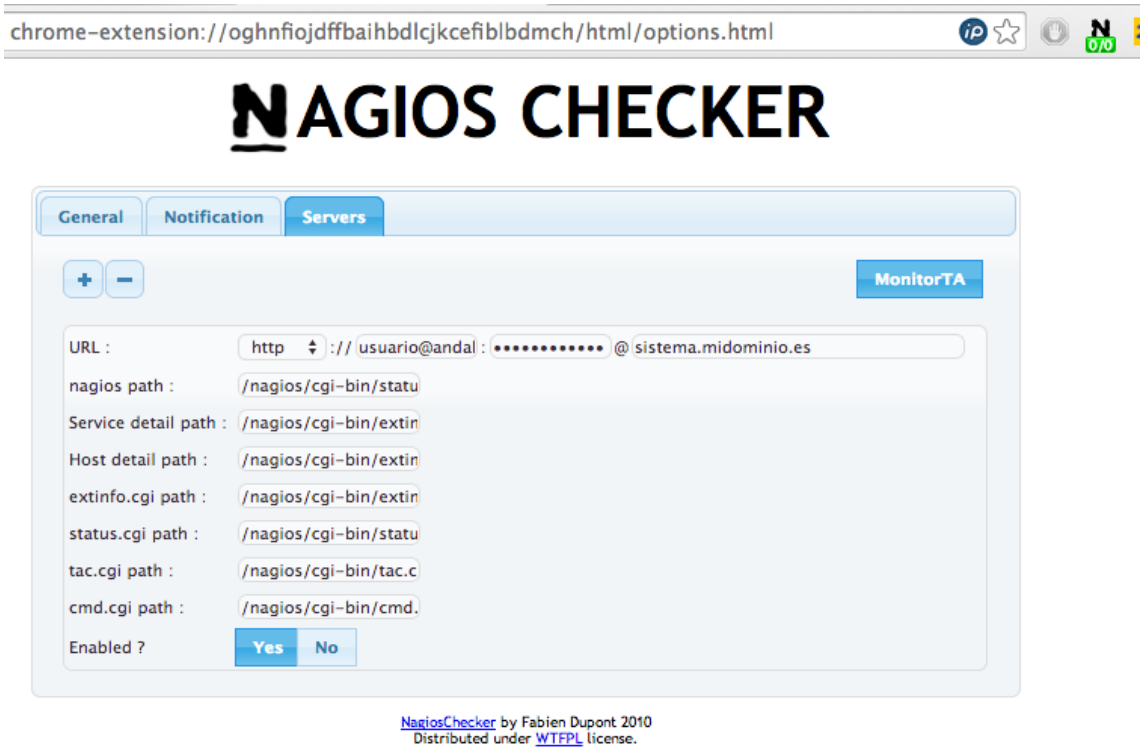


Figura 3.9. Nagios Checker para Google Chrome

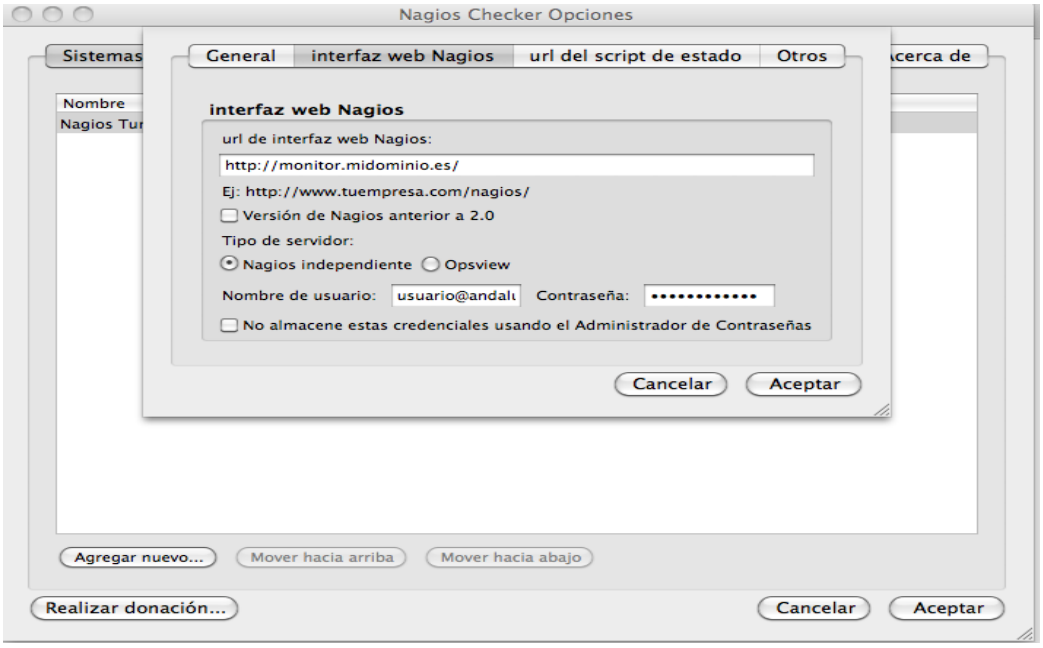


Figura 3.10. Nagios Checker para Mozilla Firefox

3.5.2. Aplicaciones móviles

Actualmente existen numerosas aplicaciones para instalar en nuestros teléfonos móviles inteligentes o en tablets. Dependiendo del servicio e información que nos proporcionen, serán gratuitas o de pago. Al igual que en el caso de los complementos en navegadores, tras proporcionarle los datos de configuración para poder acceder a nuestro sistema de monitorización, en tiempo real, tendremos toda la información de nuestro sistemas y podremos establecer nuevos sistemas de notificaciones como sonidos.

A continuación, se enumeran las más usuales, diferenciándose por el sistema operativo del móvil en el que vayan a ser instaladas y configuradas: Android o IOS.

Android

Existen diversas opciones, pero aquí mencionaremos las dos más usuales, que pueden ser descargadas desde https://play.google.com/store, para terminales con este SO.

- Anag. **Figura 3.11**. Donde se indican datos generales del estado de la red y de las alertas producidas por cambios de estado

- Nagios+Plus. **Figura 3.12**. Muestra en qué estado se encuentran los dispositivos y podemos ampliar la información del que deseemos, incluso poner en silencio las notificaciones para este dispositivo / servicio.

IOS

Para el caso de sistemas Apple, deberemos buscar en el AppleStore aplicaciones como las siguientes. Estos ejemplos son de instalación gratuita inicialmente.

- Nagios-Mobility. **Figuras 3.13**, donde resumidamente, se muestra el estado de todos los elementos monitorizados, inidicando cuántos están bien y cuántos tienen algún tipo de alerta en este momento, tanto de servidores como de servicios. Adicionalmente, se puede consultar cada uno de ellos para ampliar la información.

- TouchMon for Nagios Lite **Figuras 3.14**, donde nos muestra varios de los servicios controlados

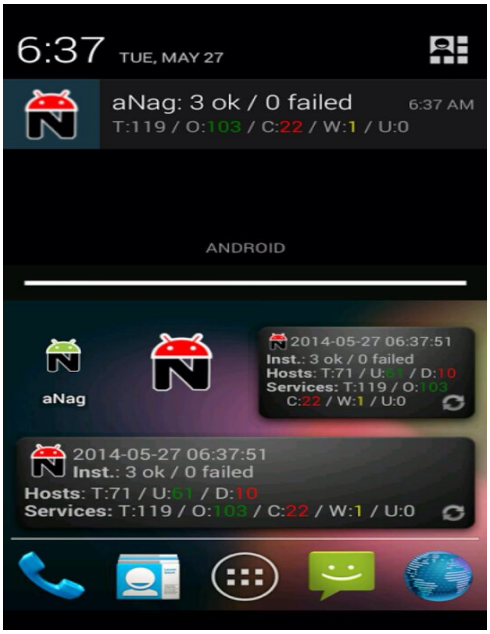


Figura 3.11. Ejemplo pantalla Anag

Figura 3.12. Ejemplo pantalla Nagios+Plus

Figura 3.13. Ejemplo pantalla Nagios-Mobility

Figura 3.14. Ejemplo pantalla TouchMon

y si con estos tipos de alertas no tenemos suficiente, siempre podremos darnos un paseo por la página http://zeldor.biz/2011/04/nagiosicinga-traffic-light/, donde nos indican cómo fabricar y configurar un auténtico semáforo que nos alertará siempre de los cambios de estado de los dispositivos que deseemos.

3.6. Aplicaciones adicionales

Como vimos en apartados anteriores, existe una gran cantidad de software disponible para el control de nuestras redes informáticas, con sus ventajas e inconvenientes, creemos conveniente detallar cómo algunas de estas herramientas pueden adicionalmente cumplimentar la labor de Nagios y aumentar el potencial de nuestra instalación de monitorización. Nos pararemos a comentar en concreto dos de ellas: cacti y mrtg.

3.6.1. Cacti

> Nagios + Cacti = monitoring, alerting, and historical performance happiness
> (Fuente: http://www.linux.com/)

Cacti, como vimos durante el comparativo de herramientas de monitorización, también es por sí sola una aplicación encargada de vigilar sistemas informáticos. Ya mencionamos que carece de algunas características que considerábamos esenciales para nuestro objetivo, sin embargo, sí que debemos tener en cuenta la completa gestión de recolección de datos y graficado que aporta respecto a los dispositivos que monitoriza. Parece entonces recomendable, que para los servidores más críticos podamos tener más información y poseer un gran histórico de la evolución de los mismos, para poder deducir qué ocurría antes de un fallo o el comportamiento irregular de algún parámetro, que por no llegar a los parámetros de umbrales de alertas, Nagios no nos va a alertar.

Además de todo esto, el propio Nagios posee un plugin que puede invocar los datos que Cacti tiene de algún dispositivo en concreto, por lo que habremos conseguido reunir una gran cantidad de información complementaria y extremadamente importante para analizar los problemas que hayan surgido y encontrar su solución de

forma más eficiente.

No nos pararemos a explicar la instalación de la herramienta ni su configuración. Pero sí quisiéramos comentar, que establecer la relación entre un dispositivo controlado por Nagios y Cacti es bastante sencilla. Partiendo de que ambas herramientas ya están funcionando correctamente, y que ambas vigilan un mismo dispositivo (misma dirección IP), bastaría con añadir una nueva línea a la definición del host dentro de Nagios, (durante la anterior descripción del fichero “hosts.cfg”, dejamos pendiente la definición de la línea correspondiente a “notes_url”):

```
define host {
        use                     generic-host
        host_name               NombreHost
        alias                   AliasHost
        address                 10.168.10.5
        icon_image              base/linux40.gif
        statusmap_image         base/linux40.png
        parents                 NombreNodoPadre
        notes_url       $USER1$/cacti/cactiplug.php?ip=$HOSTADDRESS
}
```

notes_url: este parámetro se emplea para enlazar el host con otras aplicaciones o urls, generalmente para mostrar información amplaida. En este caso invocaría la representación gráfica de sus datos correspondientes en Cacti, a través del plugin, en este caso desarrollado en “php”.

En la siguiente captura de pantalla, **Figura 3.15**, se puede apreciar cómo se accedería a esta información, desde Nagios, donde debe aparecer un enlace en forma de icono (por lo general el de Cacti, por ser lo más intuitivo) que indicará que es un dispositivo que tiene información ampliada en dentro de esta otra aplicación:

Figura 3.15. Acceso desde Nagios a Cacti

Este enlace, nos llevará dentro de la aplicación a mostrarnos la información de todos los parámetros monitorizados que posee para este dispositivo y podemos ver un ejemplo en la **Figura 3.16**.

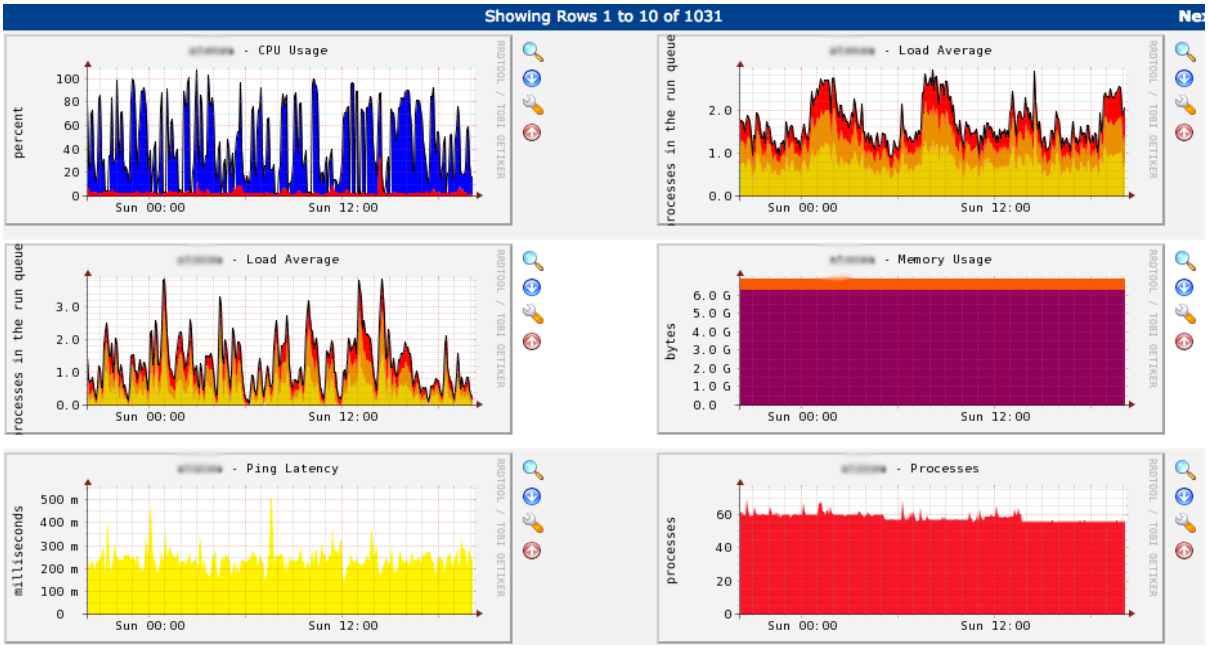


Figura 3.16. Gráficos de Cacti

A su vez, dentro de cada gráfico podremos ver el historial desde que se tiene información del servidor.

3.6.2. Mrtg

La aplicación MRTG, nos proporciona una muy detallada información, del tráfico de datos que circula por cada puerto de nuestra electrónica de red.

Hace tiempo, durante la instalación de MRTG encontré algo que resumía perfectamente la relación entre Nagios y MRTG:

Fuente: http://lena.franken.de/mrtg/

MRTG is not good for

Getting alarmed for bad values (take Nagios for that)
Getting immediate information about load of your network now

MRTG is good for

Seeing development of things over time
Looking into the past
Simple to configure! Which is good to configure things easily
Seeing load of machines, networks or services over time.

MRTG and Nagios are a good team.

Es decir, esta herramienta no nos enviará alertas, ni nos mostrará el estado actual de nuestra red, sin embargo, podremos ver lo que ha ocurrido a lo largo del tiempo y analizar cómo le afectan cambios que se vayan produciendo. Siempre, enfocado desde el punto de vista de tráfico de datos.

Al igual que ocurría en el caso anterior, esta aplicación también nos muestra información por sí sola, pero Nagios además, posee algunos plugins que nos

pueden acercar a pormenorizar detalles del tráfico de datos que se genera, como son check_mrtg (que procesa el fichero de log de MRTG y facilita la media o el mayor de los valores almacenados en él) y check_mrtgtraf (para obtener información del tráfico en un interfaz concreto).

La **Figura 3.17** muestra el contenido de alguna de estas gráficas.

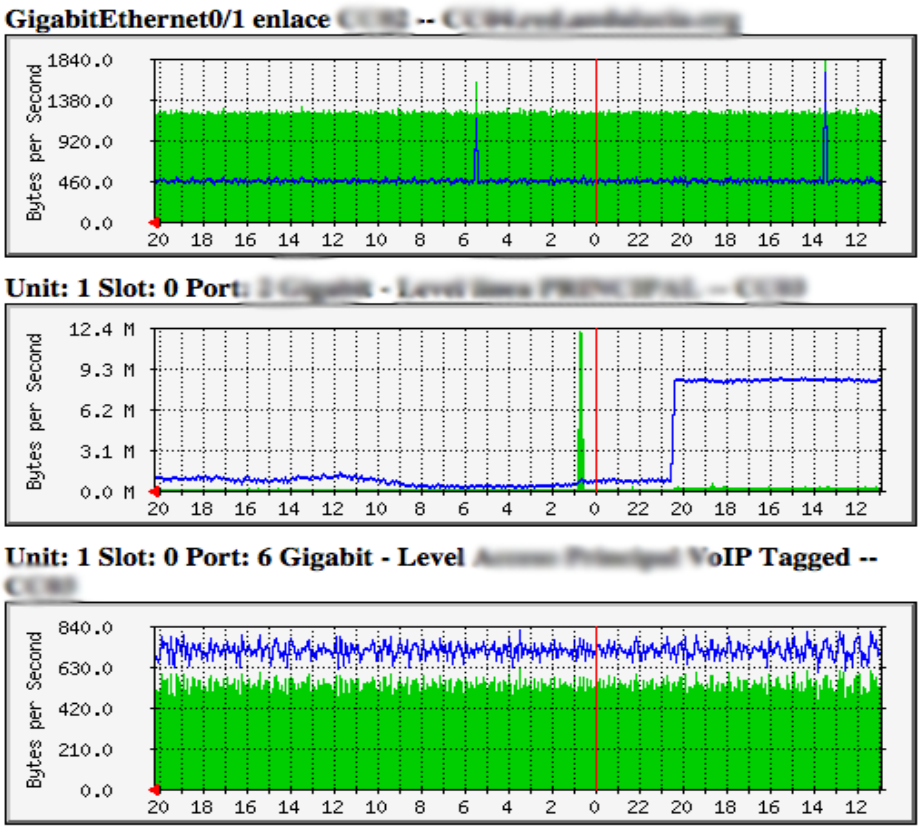


Figura 3.17. Gráficos del tráfico de datos de MRTG

4. CASO REAL EN TURISMO ANDALUZ

Llegados a este punto de la memoria del TFG, nos queda por fin mostrar el caso práctico de implementación de Nagios como herramienta de monitorización sobre una red informática ya existente, la de la Empresa Pública de Turismo Andaluz, SA. (TASA).

Como el lector, debe entender, se facilitará información de la red informática sensiblemente modificada, sobre todo en lo que se refiere a nombre de equipos y direccionamiento IP de los mismos, por medidas de seguridad obvias.

4.1. Turismo Andaluz, SA

TASA, es una empresa pública perteneciente a Consejería de Turismo y Comercio, de la Junta de Andalucía y cuyo principal objetivo es dar a conocer la marca turística de Andalucía como destino turístico a nivel internacional. Por lo que debe estar presente en multitud de eventos, ferias de promoción, etc. Además de su sede central en Málaga, posee varias subsedes en Marbella y Sevilla, y además, oficinas de turismo en las ocho provincias andaluzas.

Fue creada hace más de veinte años y ha sufrido varios traslados físicos de su infraestructura, tanto de su sede principal: primero en ubicada en Sevilla, más tarde en Marbella y actualmente en Málaga capital, como de las oficinas de turismo, que han sido trasladas a nuevas ubicaciones o se han abierto nuevas en municipios de gran interés turístico. A lo largo de su existencia, también se ha visto involucrada en procesos de fusión o división con otras empresas públicas con objetivos comunes. Por lo que se puede deducir que TASA, al estar en continua evolución provoca cambios estructurales que afectan a toda la red informática que la mantiene.

Posee un Departamento informático bien estructurado y consolidado a lo largo del tiempo, con personal especializado en distintas áreas como puede ser la microinformática, asistencia a los usuarios, aplicaciones, desarrollos web y, por último, sistemas informáticos y redes.

Se puede ampliar la información sobre TASA en http://www.andalucia.org, portal turístico de la marca Andalucía y en http://www.turismoandaluz.com, extranet de la empresa con toda su información.

4.2. Monitorización red informática TASA

Como se puede intuir, al igual que le ocurre a la mayoría de las redes informáticas de las empresas hoy día, este tipo de infraestructuras sufren cambios casi a diario y este dinamismo puede llevar a una situación de caos si no tiene “detrás” un buen sistema de monitorización siempre actualizado y bien documentado que descargue a los responsables, teniendo la seguridad de que todo está funcionando como se ha diseñado. Esta evolución puede venir tanto por cambios de ubicación, como por evolución en los equipos o sustitución de los mismos por otros más actuales y de mayor capacidad, aparición de nuevos dispositivos o servicios, etc.

Por ello, llegado el momento, se decidió llevar a cabo este trabajo de implementación de una herramienta de monitorización para toda la red informática, que acabado siendo este Trabajo Fin de Grado.

Por una parte, de forma consensuada, el departamento de sistemas analizó de forma exhaustiva todos los componentes que formaban parte de la red, agrupando sus componentes por sedes y decidiendo los dispositivos y servicios indicados para ser controlados por la monitorización, así como, la clasificación de nivel de criticidad de los mismos, contactos finales para recibir notificaciones, etc.

En el momento de la recogida de datos, están siendo monitorizados alrededor de 200 dispositivos y casi 800 servicios.

Estos dispositivos pertenecen a distintas redes con direccionamientos IP muy diferentes, por lo que en el cortafuegos se ha debido habilitar las visibilidades entre el servidor de Nagios y cada una de ellas: electrónica de red (switches y routers), sistemas de alimentación ininterrumpida (SAI), sensores ambientales o de consumo (humedad, temperatura, consumo eléctrico), redes de servidores, redes de usuarios, redes de dispositivos wifi para usuarios externos e internos, etc.

En un principio, se empezó por monitorizar todos los servidores del principal CPD con sus correspondientes servicios. Una vez comprobada la eficacia del sistema y teniendo una configuración estable, se extrapoló al resto de sedes, aunque siendo estas menos críticas, se optó porque los umbrales de alerta y parámetros de solicitud de información del estado de los elementos, se adecuaran a la importancia de cada uno de ellos.

Los dispositivos monitorizados se agruparon por nivel de importancia, similitud en su función o bien por sede, podemos ver en la siguiente **Figura 4.1** una parte de ellos:

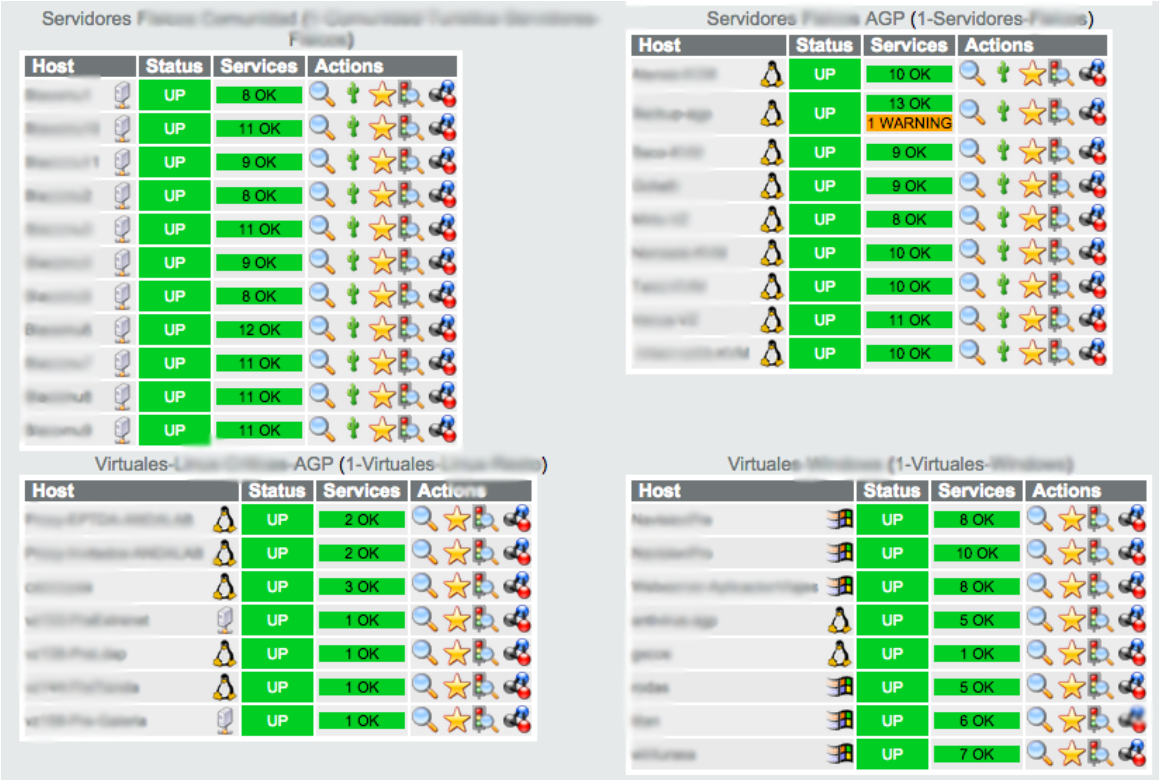


Figura 4.1. Algunos grupos de host creados para la red de TASA

En esta imagen podemos ver la información distribuida en tablas donde se han agrupado los servidores en grupos (información que previamente hemos definido en el fichero “hostgroups,cfg”) de switches/routers, servidores principales Windows o Linux, virtuales OpenVZ o KVM, sensores de consumo, etc. lo cual nos proporciona

una visión más cómoda de todos los dispositivos.

Aunque se represente de forma agrupada, en todo momento, pulsando sobre el servidor deseado, podremos acceder a toda su información o, a través de la columna “Actions”, situarlo en el mapa de la red, ir a sus datos recogidos en Cacti (si los tiene), realizar llamada al complemento PNP4Nagios y ampliar más la información almacenada.

Sin embargo, la forma más intuitiva de representar la red completa de TASA, se muestra a continuación, en la siguiente **Figura 4.2,** donde podemos ver el mapa con el que Nagios nos la representa:

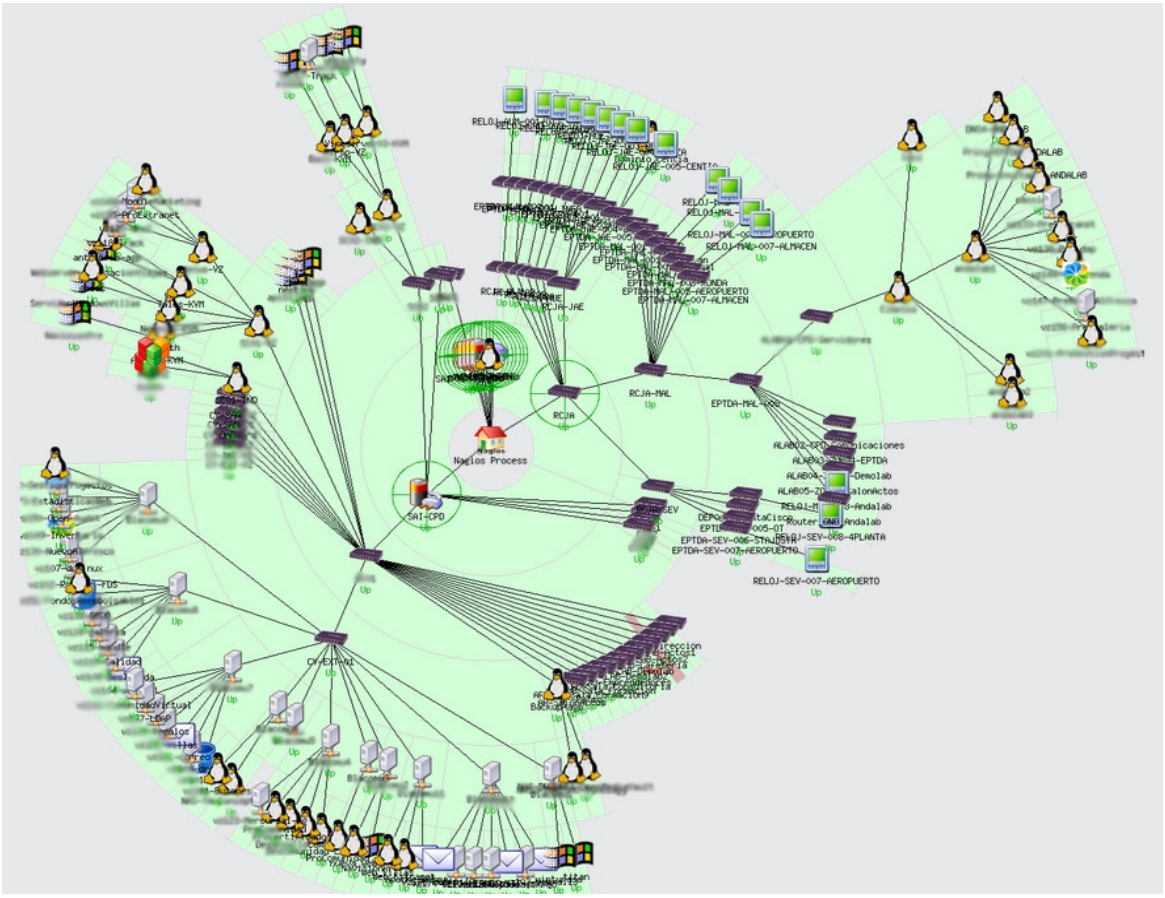


Figura 4.2. Mapa de Nagios para la red de TASA

Representación que nos será muy útil en caso de caída múltiple de servicios para ir viendo en cada momento el estado de todas las sedes o bien, para situar un

dispositivo dentro de la jerarquía que hayamos definido. Para completar esta información, sobre todo lo que está ocurriendo en la red, como comentamos durante su instalación, Nagios nos proporciona varias opciones de creación de reportes y gráficas que nos lo muestran de forma personalizada seleccionando el host/servicio, intervalos de tiempo, estados por los que ha pasado, etc. Esta información se amplía con capturas de pantalla en el apartado que veremos más adelante sobre capturas de pantallas adicionales de la aplicación.

En los siguientes apartados veremos algunos de los elementos “tipo”, que son monitorizados a través de Nagios, dependiendo del tipo de dispositivo y de su criticidad, para la red de TASA.

4.3. Servidores / Servicios monitorizados en TASA

En este apartado veremos cómo se monitorizan algunos los elementos de TASA.

Servidor OpenVZ

Host	Service	Status	Last Check	Duration	Attempt	Status Information
[redacted]	Amanda-OK	OK	09-21-2014 20:42:18	4d 22h 32m 34s	1/3	FILE_AGE OK: /var/lib/amanda/amandates is 59267 seconds old and 364 bytes
	Carga Actual	OK	09-21-2014 20:42:36	4d 22h 32m 10s	1/3	OK - load average: 0.45, 0.80, 0.92
	Dumps-OK	OK	09-21-2014 20:40:56	4d 22h 33m 49s	1/3	FILE_AGE OK: /vz/dump is 4101 seconds old and 4096 bytes
	Espacio en Boot	OK	09-21-2014 20:35:26	4d 22h 29m 19s	1/3	DISK OK - free space: /boot 58 MB (31% inode=99%):
	Espacio en Raiz	OK	09-21-2014 20:43:56	4d 12h 36m 45s	1/3	DISK OK - free space: / 3611 MB (75% inode=94%):
	Espacio_en_vz	OK	09-21-2014 20:42:36	4d 22h 32m 10s	1/3	DISK OK - free space: /vz 152262 MB (53% inode=93%):
	Espacio_en_vz_dump	OK	09-21-2014 20:43:26	4d 22h 31m 25s	1/3	DISK OK - free space: /vz/dump 391242 MB (81% inode=99%):
	PING	OK	09-21-2014 20:35:26	4d 22h 39m 19s	1/3	PING OK - Packet loss = 0%, RTA = 0.23 ms
	Procesos Activos	OK	09-21-2014 20:37:55	4d 22h 36m 50s	1/3	PROCS OK: 540 processes
	Procesos Zombies	OK	09-21-2014 20:42:36	4d 22h 32m 10s	1/3	PROCS OK: 0 processes with STATE = Z
	Usuarios	OK	09-21-2014 20:35:06	4d 22h 29m 40s	1/3	USERS OK - 0 users currently logged in

Figura 4.3. Monitorización Servidor OpenVZ

Como comentábamos, no todos los servidores son “vigilados” de la misma forma, así los servicios a tener en cuenta en cada uno de ellos pueden ser diferentes. Para este caso, según vemos en la **Figura 4.3**, por ser un servidor físico para dar servicio a virtualización OpenVZ, se monitorizan servicios a través del cliente NRPE como:

Amanda-OK: si se ha realizado el proceso de copias de seguridad del sistema correctamente. Se comprueba que existe un fichero en el directorio indicado con una antigüedad inferior a 24 horas.

Carga Actual: se comprueba el estado de la carga del sistema

Dumps-OK: indica que se han realizado correctamente las copias (o dumps) de los

contenedores virtuales que alberga en ese momento

Espacio en Boot / Raíz: el espacio en la partición de arranque o raíz del sistema es correcta.

Espacio_en_vz: El espacio destinado a los contenedores virtuales es correcto

Espacio:en_vz_dump: El espacio para las copias de seguridad de los servidores virtuales es suficiente por el momento.

Procesos Activos: el número de procesos activos para que el servidor pueda gestionarlos es inferior a los límites establecidos

PING: La respuesta al ping (icmp) está dentro de los umbrales de tiempo establecidos. Sería el único servicio, en este caso, que no sería un chequeo pasivo, a la espera de que el cliente NRPE le responda.

Procesos zombies: el número de procesos zombies no supera el límite indicado

Usuarios: el número de usuarios conectados simultáneamente al servidor no supera el mínimo para enviar notificaciones y cambiar su estado.

Podemos comprobar, que por ser un servidor crítico, tiene enlace a Cacti para ampliar su información de forma gráfica más precisa y al complemento PNP4Nagios, que nos aportará más datos del comportamiento en todo el período del que se tiene información y que será visto con algo más de detalle más adelante.

Servidor de aplicaciones Web

En este caso, también se monitorizan aspectos de funcionamiento del propio servidor, pero además se tienen en cuentan otros aspectos fundamentales para el

funcionamiento de un servidor cuya principal función es servir un portal web, como vemos en la **Figura 4.4**:

Host	Service	Status	Last Check	Duration	Attempt	Status Information
[redacted]	Carga Actual	OK	09-21-2014 20:43:45	4d 22h 35m 36s	1/3	OK - load average: 2.03, 2.39, 2.49
	Error 500 www [redacted]	OK	09-21-2014 20:44:25	1d 15h 14m 57s	1/3	HTTP OK: HTTP/1.1 302 FOUND - 362 bytes in 1.621 second response time
	Espacio en Raiz	OK	09-21-2014 20:41:05	4d 22h 38m 17s	1/3	DISK OK - free space: / 49975 MB (27% inode=27%):
	Inodos	OK	09-21-2014 20:47:18	4d 22h 32m 4s	1/3	DISK OK - free space: / 49967 MB (27% inode=27%):
	PING	OK	09-21-2014 20:47:48	4d 22h 35m 36s	1/3	PING OK - Packet loss = 0%, RTA = 0.23 ms
	Procesos Activos	OK	09-21-2014 20:46:35	4d 22h 32m 50s	1/3	PROCS OK: 41 processes
	Procesos Zombies	OK	09-21-2014 20:42:55	4d 22h 36m 31s	1/3	PROCS OK: 0 processes with STATE = Z
	Puerto 5432 - Postgres	OK	09-21-2014 20:47:18	4d 22h 38m 2s	1/3	TCP OK - 0.000 second response time on port 5432
	Puerto 8080	OK	09-21-2014 20:45:49	4d 22h 35m 36s	1/3	TCP OK - 0.000 second response time on port 8080
	Usuarios	OK	09-21-2014 20:46:35	4d 22h 32m 50s	1/3	USERS OK - 0 users currently logged in

Figura 4.4. Monitorización Servidor de aplicaciones Web

Error 500 www.midominio.es: Controlar que el servidor web no esté produciendo un error 500, lo cual indica que no se puede acceder a la página principal del portal.

Inodos: el número de inodos, cuando es un sistema que gestiona muchos ficheros también se deberá tener en cuenta por si se debe ampliar este recurso del servidor.

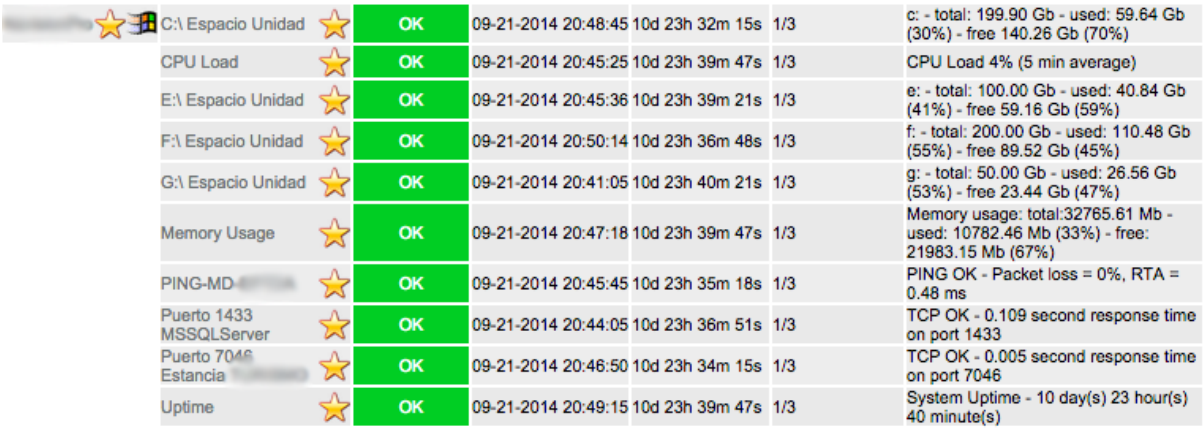
Puerto 5432 – Postgres: Comprobar que el puerto que debe aceptar peticiones a la base de datos, en este caso Postgres, está escuchando correctamente.

Puerto 8080: Cualquier otro puerto, en este caso el 8080, puede comprobarse que está activo, por si a través de él se gestiona alguna administración.

Servidor Windows

A través del cliente NSClient++, según la Figura 4.5, obtendremos respuesta a los servicios que nos interesen más en este tipo de servidores Windows, alguno de

ellos, similares a los que poseen sistema operativo Linux, como en los anteriores casos, con la diferencia que aquí los sistemas de archivos serán identificados como unidades de red. El resto de aspectos como carga, uso de memoria o cualquier tipo de puerto, se realiza de la misma forma.



C:\ Espacio Unidad	OK	09-21-2014 20:48:45	10d 23h 32m 15s	1/3	c: - total: 199.90 Gb - used: 59.64 Gb (30%) - free 140.26 Gb (70%)
CPU Load	OK	09-21-2014 20:45:25	10d 23h 39m 47s	1/3	CPU Load 4% (5 min average)
E:\ Espacio Unidad	OK	09-21-2014 20:45:36	10d 23h 39m 21s	1/3	e: - total: 100.00 Gb - used: 40.84 Gb (41%) - free 59.16 Gb (59%)
F:\ Espacio Unidad	OK	09-21-2014 20:50:14	10d 23h 36m 48s	1/3	f: - total: 200.00 Gb - used: 110.48 Gb (55%) - free 89.52 Gb (45%)
G:\ Espacio Unidad	OK	09-21-2014 20:41:05	10d 23h 40m 21s	1/3	g: - total: 50.00 Gb - used: 26.56 Gb (53%) - free 23.44 Gb (47%)
Memory Usage	OK	09-21-2014 20:47:18	10d 23h 39m 47s	1/3	Memory usage: total:32765.61 Mb - used: 10782.46 Mb (33%) - free: 21983.15 Mb (67%)
PING-MD	OK	09-21-2014 20:45:45	10d 23h 35m 18s	1/3	PING OK - Packet loss = 0%, RTA = 0.48 ms
Puerto 1433 MSSQLServer	OK	09-21-2014 20:44:05	10d 23h 36m 51s	1/3	TCP OK - 0.109 second response time on port 1433
Puerto 7046 Estancia	OK	09-21-2014 20:46:50	10d 23h 34m 15s	1/3	TCP OK - 0.005 second response time on port 7046
Uptime	OK	09-21-2014 20:49:15	10d 23h 39m 47s	1/3	System Uptime - 10 day(s) 23 hour(s) 40 minute(s)

Figura 4.5. Monitorización Servidor Windows

Dispositivo tipo switch

En este caso, en la **Figura 4.6**,el dato más importante, a parte de que esté encendido, será controlar que existe tráfico en los puertos del switch/router que podremos identificar con la etiqueta de la tarjeta de red o dispositivo del que deba proceder dicho intercambio de datos. En el caso del ejemplo, se comprueba que hay tráfico tanto de la Línea principal de la sede como la de respaldo, o bien, en la conexión con otro switch/router, así como, un enlace secundario.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
	- Conexion con	OK	09-21-2014 20:51:05	584d 5h 23m 19s	1/3	SNMP OK - up(1)
	- Conexion con	OK	09-21-2014 20:53:15	584d 5h 20m 51s	1/3	SNMP OK - up(1)
	- Conexion con	OK	09-21-2014 20:55:45	584d 5h 18m 23s	1/3	SNMP OK - up(1)
	- Enlace con	OK	09-21-2014 20:50:27	27d 7h 49m 1s	1/3	SNMP OK - up(1)
	- Linea Backup	OK	09-21-2014 20:51:05	124d 1h 1m 51s	1/3	SNMP OK - up(1)
	- Linea principal	OK	09-21-2014 20:53:15	123d 22h 42m 51s	1/3	SNMP OK - up(1)
	PING	OK	09-21-2014 20:56:56	9d 8h 2m 33s	1/3	PING OK - Packet loss = 0%, RTA = 1.06 ms
	Tiempo Actividad	OK	09-21-2014 20:50:27	27d 7h 49m 1s	1/3	SNMP OK - Timeticks: (854780404) 98 days, 22:23:24.04

Figura 4.6. Monitorización dispositivo tipo Switch

Sistema de copias de seguridad

Host	Service	Status	Last Check	Duration	Attempt	Status Information
	Amanda-OK	OK	09-21-2014 20:58:26	60d 21h 34m 36s	1/3	FILE_AGE OK: s 71764 seconds old and 308 bytes
	Carga Actual	OK	09-21-2014 21:00:42	72d 9h 8m 13s	1/3	OK - load average: 0.04, 0.01, 0.00
	Copia_bbdd_	WARNING	09-21-2014 20:59:16	0d 20h 3m 37s	3/3	FILE_AGE WARNING: is 158955 seconds old and 12288 bytes
	Copia_bbdd_	OK	09-21-2014 20:54:05	1d 20h 6m 49s	1/3	FILE_AGE OK: is 72183 seconds old and 4096 bytes
	Espacio en Boot	OK	09-21-2014 21:00:27	52d 1h 10m 35s	1/3	DISK OK - free space: /boot 25 MB (28% inode=99%):
	Espacio en Raiz	OK	09-21-2014 20:52:38	72d 9h 8m 12s	1/3	DISK OK - free space: / 4344 MB (90% inode=97%):
	Espacio_en_	OK	09-21-2014 20:51:05	72d 9h 5m 46s	1/3	DISK OK - free space: /acronis 268500 MB (56% inode=99%):
	Espacio_en_backup	OK	09-21-2014 20:59:55	52d 1h 11m 3s	1/3	DISK OK - free space: /backup 819176 MB (18% inode=99%):
	Espacio_en_copias_vz	OK	09-21-2014 20:52:26	49d 5h 28m 35s	1/3	DISK OK - free space: 521492 MB (22% inode=99%):
	Espacio_en_datos	OK	09-21-2014 20:54:51	72d 9h 8m 12s	1/3	DISK OK - free space: /datos 84244 MB (90% inode=99%):
	PING	OK	09-21-2014 20:57:18	9d 8h 3m 41s	1/3	PING OK - Packet loss = 0%, RTA = 0.24 ms
	Procesos Activos	OK	09-21-2014 20:59:55	72d 9h 13m 6s	1/3	PROCS OK: 258 processes
	Procesos Zombies	OK	09-21-2014 21:00:27	52d 1h 10m 35s	1/3	PROCS OK: 0 processes with STATE = Z
	Usuarios	OK	09-21-2014 20:56:50	72d 9h 8m 11s	1/3	USERS OK - 0 users currently logged in

Figura 4.7. Monitorización dispositivo tipo Backup

Para este tipo de servidores, una vez añadidas las alertas para los aspectos físicos, comprobaremos que se realizan las copias de las diferentes bases de datos y sistemas, así como el estado del espacio de diferentes NAS que tenga instalado y

dedicado a cada tipo de copias (**Figura 4.7**).

Terminada la monitorización de toda la red informática, incluyendo la electrónica de red también, se decidió ir añadiendo a la configuración objetos, que al ser controlados por Nagios, podía ayudar a personal de otros departamentos diferentes al informático, a desarrollar mejorar su trabajo, implementando plugins que en caso de que se produzca alguna circunstancia les envíe notificaciones que le puedan ayudar, por ejemplo, alertándolo cuando la herramienta que usan habitualmente, cree un fichero en un determinado directorio, señal de que no ha podido procesar una petición de un usuario, etc.

4.4. Ejemplos de informes presentados por Nagios

A continuación, se muestran otras capturas de pantalla del interfaz web de Nagios sobre la red de TASA, donde se muestran herramientas útiles que fueron descritas en los capítulos de instalación y configuración de forma genérica y ahora se presentan con información real.

Comportamiento de un host/servicio durante un intervalo de tiempo determinado, **Figura 4.8**, donde se puede ver que el dispositivo tuvo alguna pérdida de servicio:

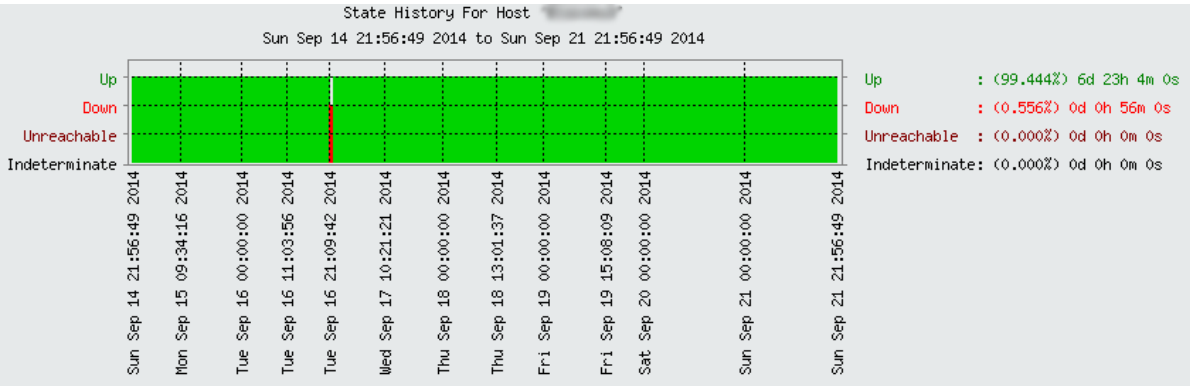


Figura 4.8. Ejemplo de comportamiento de dispositivo en un período de tiempo concreto

Podemos ver un informe de alertas en la **Figura 4.9**, resumen de todas las alertas enviadas, clasificadas por servidores ó servicios, así como por el estado en el que se encontraban cuando se enviaron.

Host Alerts

State	Soft Alerts	Hard Alerts	Total Alerts
UP	39	2	41
DOWN	51	2	53
UNREACHABLE	21	1	22
All States	111	5	**116**

Service Alerts

State	Soft Alerts	Hard Alerts	Total Alerts
OK	117	35	152
WARNING	102	26	128
UNKNOWN	0	0	0
CRITICAL	80	1311	1391
All States	299	1372	**1671**

Figura 4.9. Resumen de alertas enviadas por el sistema

Por otro lado, se puede obtener un informe de cada una de las notificaciones

enviadas, según vemos en la **Figura 4.10**.

Host	Service	Type	Time	Contact	Notification Command	Information
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:30	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:30	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:30	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:30	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:23	[redacted]	notify-host-by-sms	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:16	[redacted]	notify-host-by-sms	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:09	[redacted]	notify-host-by-sms	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 22:26:02	[redacted]	notify-host-by-sms	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:26:02	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:26:02	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:26:02	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:26:02	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:24:52	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:24:52	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:24:52	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable
[redacted]	N/A	HOST DOWN	09-15-2014 21:24:52	[redacted]	notify-host-by-email	CRITICAL - Host Unreachable

Figura 4.10. Distintos tipos de alertas enviadas sms/email

La cola en la que se sucederán los próximos chequeos, y cómo podemos actualizarla en caso de que sea necesario, **Figura 4.11**.

Entries sorted by **next check time** (ascending)

Host	Service	Last Check	Next Check	Type	Active Checks	Actions
[redacted]	[redacted]	09-09-2014 10:14:52	09-09-2014 10:24:52	Normal	ENABLED	
[redacted]		09-09-2014 10:19:45	09-09-2014 10:24:54	Normal	ENABLED	
[redacted]		09-09-2014 10:19:45	09-09-2014 10:24:54	Normal	ENABLED	
[redacted]	Inodos	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Procesos Zombies	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Espacio en Raiz	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Procesos Zombies	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Espacio en Raiz	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Procesos Zombies	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Espacio en Raiz	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Memory Usage	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	PING-Reloj	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Espacio en Raiz	09-09-2014 10:14:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	PING-MD-LAB	09-09-2014 10:19:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	PING-MD-LAB	09-09-2014 10:19:56	09-09-2014 10:24:56	Normal	ENABLED	
[redacted]	Memory Usage	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Usuarios	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Inodos	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Procesos Zombies	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Usuarios	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Procesos Zombies	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Espacio_en_datos	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	Usuarios	09-09-2014 10:15:02	09-09-2014 10:25:02	Normal	ENABLED	
[redacted]	C:\ Espacio Unidad	09-09-2014 10:15:03	09-09-2014 10:25:03	Normal	ENABLED	
[redacted]		09-09-2014 10:19:54	09-09-2014 10:25:04	Normal	ENABLED	
[redacted]		09-09-2014 10:19:54	09-09-2014 10:25:04	Normal	ENABLED	
[redacted]	Espacio_en_acronis	09-09-2014 10:15:09	09-09-2014 10:25:09	Normal	ENABLED	

Figura 4.11. Cola de próximos chequeos de Nagios

4.5 Tipos de alertas/notificaciones recibidas

Como vimos durante la configuración de Nagios, las notificaciones le llegarán a los contactos correspondientes, una vez que se produzca un cambio de estado. En el caso de TASA, se han realizado distintos grupos de contactos que recibirán las alertas cuando esté implicado un dispositivo de su responsabilidad. Por defecto, se enviarán por email tanto el error como cuando vuelva a un estado correcto. En aquellos casos que los dispositivos o servicios sean de vital importancia para el funcionamiento de la empresa, se enviarán SMS. En las siguientes imágenes se muestran ejemplos de alertas enviadas por ambas vías:

Email

En el sección 3.3.8. templates.cfg, ya vimos cómo se definían las plantillas para personalizar el tipo de correos que deseamos enviar. En nuestro caso, creemos que

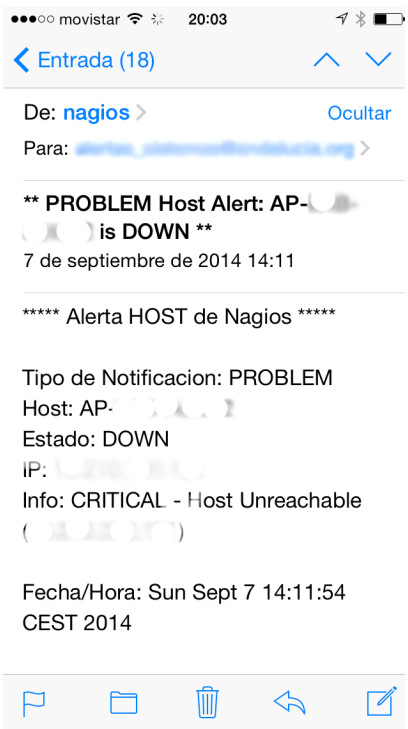


Figura 4.12. Ejemplo Email 1

Figura 4.13. Ejemplo Email 2

deben ser escuetos y presentar la información justa y necesaria para, de un vistazo poder conocer el tipo de problema que ha surgido. En las siguientes capturas, se muestran ejemplos de notificación de un problema y de una recuperación.

Vemos algunas capturas de pantalla en las **Figuras 4.12** y **4.13**. La primera, muestra un email de alerta por caída de un AP wifi y la segunda, la vuelta a un estado correcto, de la “Carga Actual” de un servidor. En ambos, se indica la fecha y hora en la que ocurrió el cambio de estado.

SMS

Para el caso de los SMS, los textos son más cortos aún, pero igualmente, con la información necesaria para saber cuál ha sido el problema que se ha producido. Vemos en la siguiente **Figura 4.14**, donde se muestran varios SMS de error y de recuperación de diferentes dispositivos.

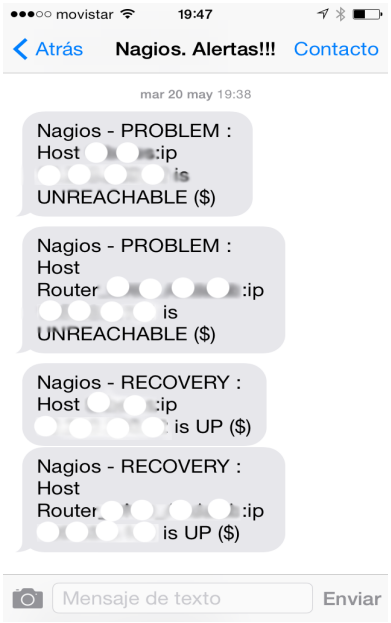


Figura 4.14. Ejemplo SMS

4.6. Complementos añadidos

La comunidad formada por todos los usuarios de Nagios sigue de forma continua añadiendo complementos que mejoran la herramienta y presenta elementos útiles para completar nuestra instalación. Algunos de los añadidos a la configuración básica de TASA, son los siguientes:

NagMap: complemento para posicionar las diferentes sedes de TASA sobre los mapas google para tener mejor comprensión geográficas de las mismas. En la **Figura 4.15**, se muestra un ejemplo. En la columna de la derecha (por motivos de seguridad contiene los datos difuminados) muestra el estado de cada sede y de los dispositivos de cada una de ellas.

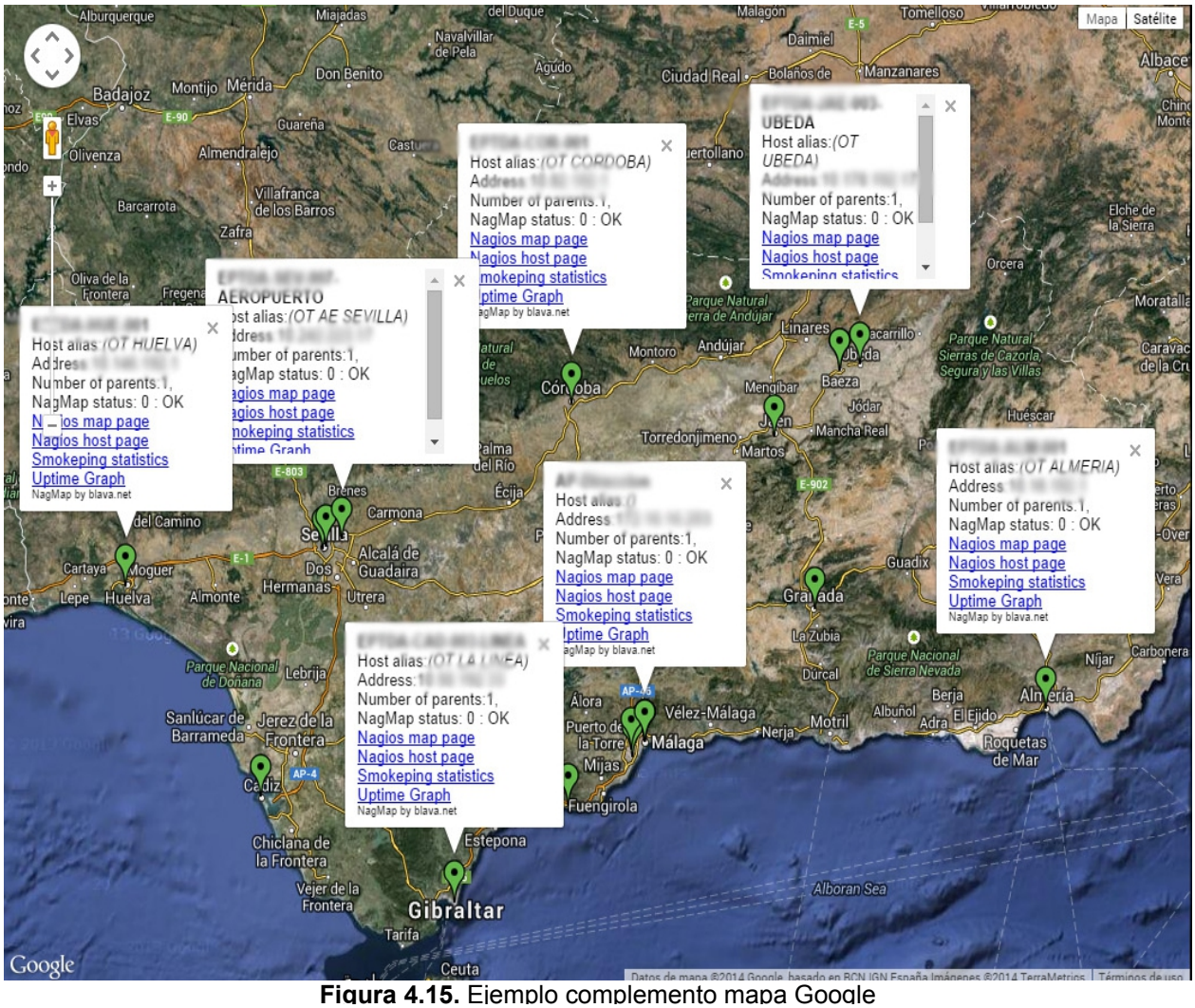


Figura 4.15. Ejemplo complemento mapa Google

PNP4Nagios: complemento que nos proporciona gráficas básicas del comportamiento de los objetos monitorizados. Consiste en añadir una llamada desde servidor/servicio deseado (en la siguiente **Figura 4.16** se realiza mediante un icono en forma de estrella) al complemento que nos muestra la información que contiene hasta este momento y cómo se visualiza en la **Figura 4.17**:

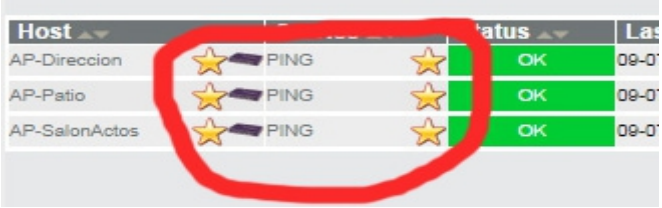


Figura 4.16. Ejemplo llamada al complemento PNP4Nagios

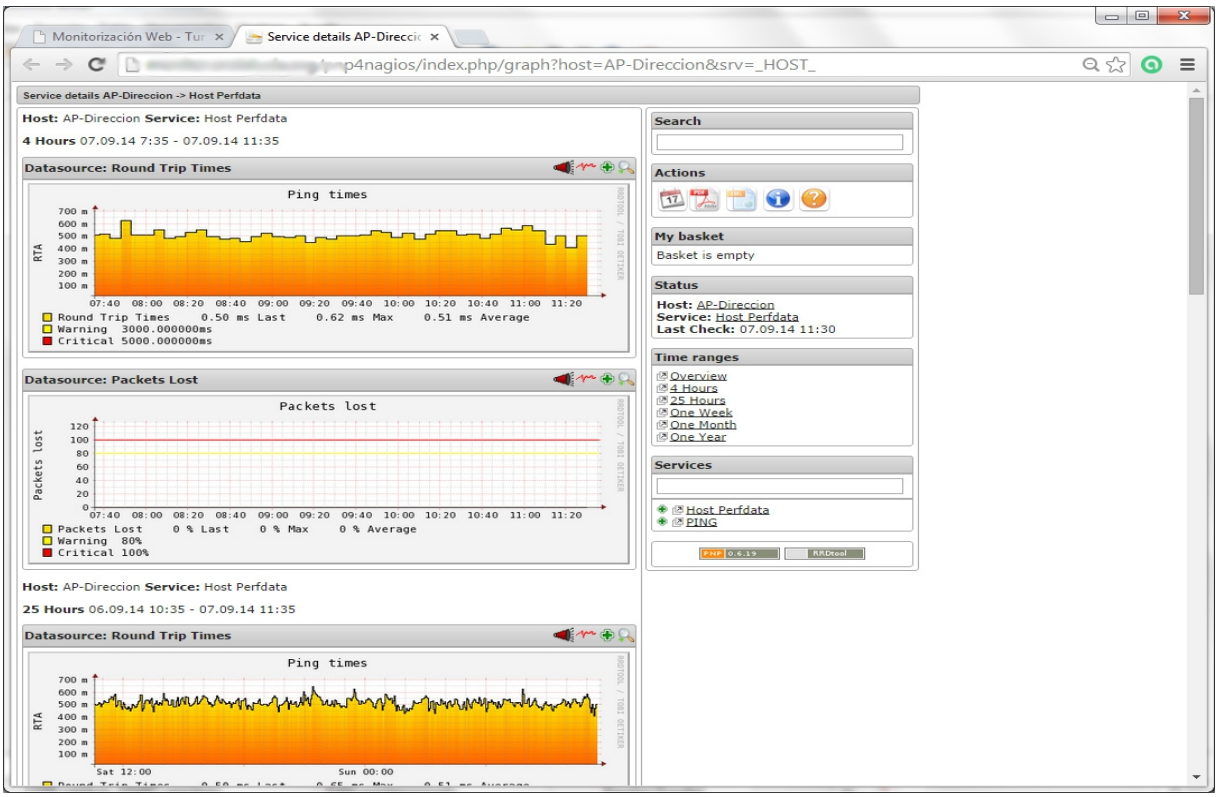


Figura 4.17 Ejemplo de gráficas del complemento PNP4Nagios

Conclusiones

Como ocurre muchas veces en la vida, la necesidad es la que nos obliga a buscar soluciones a los nuevos problemas que surgen a diario. La instalación de un tipo de herramientas como Nagios sobre la red informática de TASA, ha sido como encontrar la cuadratura del círculo. Actualmente, existe una red bastante compleja y se ha conseguido que sea estable, segura y fiable, por lo que el siguiente paso debía ser añadir un sistema que la monitorice y nos avise de aquello que consideramos un problema, error o simplemente, algo que no está funcionando correctamente.

Tras haber realizado la implementación de la misma y llegado a un sistema de monitorización robusto, y sobre casi todos los recursos disponibles, hemos conseguido dedicar tiempo del departamento a otras funciones. Por otra parte, no menos importante, al ser alertados en cuanto se produce un problema, la capacidad de reacción es inmediata y, a la vez que se corrigen en menos tiempo, evitamos que se produzcan otros errores mayores. Desde luego, ha sido una de las aplicaciones que más ha colaborado al buen desarrollo del trabajo en el departamento informático de TASA.

Por otra parte, se ha constatado que este aplicación está tan “viva” como la red que vigila, ya que casi a diario, aparecen nuevos elementos que deben ser monitorizados, o servicios que debemos añadir a los dispositivos ya existentes, así como, la incorporación de nuevos tipos de hardware y software, para los cuales en caso de no existir forma de vigilar su estado, se intentan desarrollar aplicaciones para ello. De hecho, el empleo de esta herramienta ha conseguido que tras surgir un problema en la red o después de que haya ocurrido alguno que era desconocido hasta ese momento, la solución para ser alertados lo antes posible, siempre pasa por implementar su control a través de Nagios y ser añadido a su configuración.

Adicionalmente, hemos añadido complementos a Nagios y empleado varias aplicaciones complementarias para poder ampliar la información de lo que estaba ocurriendo cuando se produjo algún problema o para conocer el comportamiento de un elemento desde que se decidió vigilar, reuniendo así, toda la información posible para poder tomar decisiones de forma más rápida y segura.

Para el futuro, entendemos que la herramienta debería incluir toda esta información sin que tengamos que depender de aplicaciones adicionales y sobre todo mejorar su información gráfica, la cual hemos comprobado que es bastante elemental.

La principal conclusión a la que se llega tras la realización de este proyecto, es que durante el proceso de instalación de cualquier red informática, al mismo tiempo, se debería tomar la decisión sobre cuál de las herramientas de monitorización de redes, es la que mejor se adapta al tipo de red que se va a configurar, teniendo en cuenta con toda seguridad, que tanto el conjunto, como los elementos vigilados estarán en continua evolución y necesitará estar bien implementada para proporcionar seguridad a los responsable de la misma.

Referencias bibliográficas

- Wikipedia: Imágenes y entradas sobre monitorización de redes y herramientas.

- Documento "3-Doc.Estado del Arte - Comparativo herramientas.pdf": Archivo de referencia para el comparativo de herramientas de monitorización.

- http://www.nagiosexchange.org/: Página de la Comunidad Nagios

- http://www.nagios.org/: Página oficial de Nagios

- http://nagios.sourceforge.net/: Principal repositorio de Nagios

- http://www.cacti.net/: Página oficial de Cacti

- http://oss.oetiker.ch/mrtg/: Página oficial de MRTG

- http://es.wammu.eu/smsd/ : Página oficial de Gammu

Anexo 1. Plugins de Nagios

check_alfresco
check_dhcp
check_file_exists
check_http
check_jabber
check_mem.pl
check_nntp
check_nwstat
check_radius
check_snmp
check_udp
check_alfresco.jar
check_dig
check_flexlm
check_icmp
check_kvm
check_mrtg
check_nntps
check_oracle
check_real
check_spop
check_ups
check_apt
check_disk
check_fping
check_ide_smart
check_ldap
check_mrtgtraf
check_nrpe
check_overcr
check_rpc
check_ssh
check_users
check_breeze
check_disk_smb
check_ftp
check_ifoperstatus
check_ldaps
check_mssql.php
check_nt
check_pgsql
check_samba
check_ssmtp
check_virsh
check_by_ssh
check_dns
check_game
check_ifstatus
check_load
check_mysql
check_ntp
check_ping
check_sensors
check_swap
check_virt
check_clamd
check_dummy
check_hddtemp.sh
check_imap
check_log
check_mysql_query
check_ntp_peer
check_pop
check_simap
check_tcp
check_wave
check_cluster
check_file_age
check_hpjd
check_ircd
check_mailq
check_nagios
check_ntp_time
check_procs
check_smtp
check_time

Anexo 2. Ejemplos de plantillas

Se muestran a continuación, detalles de definición de distintos tipos de plantillas. Una información más completa de todos los parámetros y sus posibles valores, pueden encontrarse en http://nagios.sourceforge.net/.

Plantilla para contactos que deban recibir notificaciones mediante SMS, durante ciertas horas del día.

```
#-----------------------------------------------------------
# Plantilla de contactos con SMS
#-----------------------------------------------------------

define contact{
    name                            sms             ; The name of this contact template
    service_notification_period     smshoras        ; service notifications can be sent anytime
    host_notification_period        smshoras        ; host notifications can be sent anytime
    service_notification_options    w,u,c,r,f,s     ; send notifications for all service states,
                                                    flapping events, and scheduled downtime events
    host_notification_options       d,u,r,f,s       ; send notifications for all host states,
                                                    flapping events, and scheduled downtime events
    service_notification_commands   notify-service-by-sms ; send service notifications via email
    host_notification_commands      notify-host-by-sms    ; send host notifications via email
    register                        0       ; DONT REGISTER THIS DEFINITION -
                                            ITS NOT A REAL CONTACT, JUST A TEMPLATE!
    }
```

Plantilla para hosts genéricos, que deben estar continuamente monitorizados: se especifica que puede realizar envíos de alertas o ejecutar eventos, tiempos entre chequeos, etc.

```
#-------------------------------------------------------------------------
# Plantilla de la mayoría de los host, definidos en hosts.cfg
#-------------------------------------------------------------------------
define host{
        name                            generic-host ; Nombre de la Plantilla
        check_period                    24x7
        notifications_enabled           1 ; Notificaciones del host habilitadas (0,1)
        event_handler_enabled           1 ; Event Handler habilitado (0,1)
        flap_detection_enabled          1 ; Flap Detection habilitado (0,1)
        failure_prediction_enabled      1 ; Failure prediction Habilitado (0,1)
        process_perf_data               1 ; Processar datos de estado (0,1)
        retain_status_information       1 ; Mantener datos entre restarts (0,1)
        retain_nonstatus_information    1 ; Mantener non-status entre restarts
        notification_period             24x7 ; Enviar notificaciones todos los dias.
        check_interval                  5               ; Actively check the host every 5 minutes
        retry_interval                  1               ; Schedule host check retries at 1 minute intervals
        check_command                   check-host-alive ; Default command to check Linux hosts
        notification_interval           120             ; Resend notifications every 2 hours
        max_check_attempts              10          ; Máximo veces intentara probar un check
        notification_options            d,u,r ;
        contact_groups                  nagiosadmin
            register                    0               ; DONT REGISTER THIS DEFINITION - ITS
                                                          NOT A  REAL HOST, JUST A TEMPLATE!
}

#-------------------------------------------------------------
# Plantilla dispositivos para control de presencia
#-------------------------------------------------------------

define host{
```

```
        name                presencia                ; The name of this host template
        use                 generic-host             ; This template inherits other values
                                                     from the generic-host template
        contact_groups      nagiosadmin_presencia    ; Notifications get sent to the admins by default
        }

define host{
        name                generic-switch           ; The name of this host template
        use                 generic-host             ; Inherit default values from the generic-host template
        check_period        24x7                     ; By default, switches are monitored round the clock
        check_interval      5                        ; Switches are checked every 5 minutes
        retry_interval      1                        ; Schedule host check retries at 1 minute intervals
        max_check_attempts  10                       ; Check each switch 10 times (max)
        check_command       check-host-alive         ; Default command to check if routers are "alive"
        notification_period 24x7                     ; Send notifications at any time
        notification_interval 60                     ; Resend notifications every 60 minutes
        notification_options d,r                     ; Only send notifications for specific host states
        contact_groups      nagiosadmin              ; Notifications get sent to the admins by default
        register            0                        ; DONT REGISTER THIS - ITS JUST A TEMPLATE
}

#-------------------------------------------------------
# Plantilla general de Servicios
#-------------------------------------------------------

define service{
        name                    generic-service      ; The 'name' of this service template
        active_checks_enabled   1                    ; Active service checks are enabled
        passive_checks_enabled  1                    ; Passive service checks are enabled/accepted
        parallelize_check       1                    ; Active service checks should be parallelized
                                                     (disabling this can lead to major performance problems)
        obsess_over_service     1                    ; We should obsess over this service (if necessary)
```

```
    #check_freshness                0               ; Default is to NOT check service 'freshness'
    notifications_enabled           1               ; Service notifications are enabled
    event_handler_enabled           1               ; Service event handler is enabled
    flap_detection_enabled          1               ; Flap detection is enabled
    failure_prediction_enabled      1       ; Failure prediction is enabled
    process_perf_data               1       ; Process performance data
    retain_status_information       1       ; Retain status information across program restarts
    retain_nonstatus_information    1       ; Retain non-status information across program restarts
    is_volatile                     0       ; The service is not volatile
    check_period                    24x7    ; The service can be checked at any time of the day
max_check_attempts                  3       ; Re-check the service up to 3 times in order to
                                            determine its final (hard) state
normal_check_interval               10      ; Check the service every 10 minutes under normal
                                             conditions
retry_check_interval                2       ; Re-check the service every two minutes until a hard
                                            state can be determined
 contact_groups                     nagiosadmin  ; Notifications get sent out to everyone in the
                                             'admins' group
 notification_options               w,u,c,r ; Send notifications about warning, unknown, critical,
                                             and recovery events
notification_interval               120     ; Re-notify about service problems every hour
notification_period                 24x7    ; Notifications can be sent out at any time
register                            0       ; DONT REGISTER THIS DEFINITION - ITS NOT A REAL
                                            SERVICE, JUST A TEMPLATE!

    }
```